Universidad Carlos III de Madrid
Departamento de Informática

Proyecto de fin de carrera

Estudio e implementación de algoritmos genéticos para la generación semi-automática de ritmos y melodías

Autor: Enrique Jiménez Domingo
Tutor: Yago Sáez Achaerandio
Co-director: Pedro Isasi Viñuela

Leganés, 2009

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

El universo de la Inteligencia Artificial es uno de los campos más interesantes y amplios del mundo de la Informática. Sus técnicas, potencia y el gran abanico de áreas de aplicación que abarca, hacen de ella una de las principales fuentes de investigación y experimentación, además de servir como punto de referencia para lograr avances y mejoras en todos los campos imaginables.

Dentro de ese gran abanico no sólo se encuentran trabajos directamente relacionados con la informática, sino que también existen aplicaciones en el mundo empresarial (gestión del tiempo, gestión de recursos, optimización de costes, análisis bursátil...), en juegos, medicina e incluso en las artes. Éste último siempre ha sido un espacio complicado y apasionante en el que, aunque a priori parecía complicado introducirse, se han realizado ya varias incursiones de gran interés. Por ello, avances que hasta hace unos años se antojaban impensables, hoy en día son una realidad. Actualmente, podemos encontrar desde pequeños robots capaces de pintar ([Moura, 2004]), a otros con la habilidad de escribir ([Moura, 2005]) e incluso los primeros sistemas capaces de expresar sensibilidad y emociones ([Shibata and Jennings, 1996]).

Otra rama artística que ha sido tratada por la informática es la de la música. Como todos sabemos, la utilización del ordenador en la creación musical es un fenómeno extendido. A medida que se han sucedido los avances en este campo los ordenadores personales se han ido conviertiendo en una herramienta cada vez más indispensable ya que permiten introducir multitud de efectos y giros en melodías y ritmos, simplificar el proceso de mezclado de pistas y facilitar el acceso a la información musical propiamente dicha.

Sin embargo, esta parte de la informática aplicada a la música carece de cualquier acercamiento a la Inteligencia Artificial ya que, en este caso,

lo que principalmente se hace es realizar modificaciones a obras musicales previamente creadas. Pero, ¿Y si ahora lo que se persigue es la creación de música? ¿Sería capaz un programa de componer ritmos y melodías desde cero? ¿Y utilizando técnicas de Inteligencia Artificial?

Este proyecto pretende dar respuesta a estas y otras cuestiones. A lo largo de este documento se presentará el estudio realizado sobre el empleo de técnicas de computación evolutiva a la composición de música de manera automática, incluyendo un análisis previo del problema y del entorno, el estado del arte, experimentos realizados y los resultados obtenidos para cada uno de ellos.

Uno de los factores que ha llevado a la realización de este proyecto es, como se ha comentado anteriormente, el hecho de que el área de las artes, y más concretamente el de la música, ha sido siempre uno de los campos de exploración más atractivos. Esto es debido a su gran complejidad y subjetividad que han suscitado mucho interés dentro de la comunidad informática. Aún así, los resultados obtenidos hasta la fecha no son demasiado numerosos y en un porcentaje muy elevado distan de lo que puede llegar a construir la creatividad humana. Por este motivo, el reto que plantea este proyecto es considerable, pero también otorga emoción e ilusión por dar un paso más en esta línea de investigación tan relevante.

Antes de entrar con más profundidad en las diferentes partes del estudio realizado, se presenta a continuación un breve repaso a las principales técnicas de Inteligencia Artificial existentes así como algunos conceptos que pueden resultar importantes para el completo entendimiento de los procesos realizados. A medida que se avance a través de estos aspectos, se irán relacionando con sus aplicaciones y apariciones en el mundo de la creación musical y sobre todo se analizará el motivo por el cual este área es de especial interés para el desarrollo y aplicación de Algoritmos Genéticos.

1.1. Inteligencia Artificial

Las palabras Inteligencia Artificial admiten numerosas de definiciones. Una de ellas es la de la rama de la ciencia que estudia el uso de los ordenadores para crear lo que se ha llamado "vida artificial" o "vida inteligente" ([Ray, 1992]). Lo que viene a significar esta definición es que estudia la creación de programas que a través de diferentes procesos sean capaces de evolucionar en la misma línea que lo hacen los seres humanos, naciendo, creciendo, aprendiendo, reproduciéndose y muriendo. De este modo se fusiona la capacidad de procesamiento y almacenamiento de los ordenadores con

la capacidad de razonamiento humana, lo que puede originar unos avances increíbles para la humanidad.

Ya lo dice Raymond Kurzweil ([Kurzweil, 1998]), considerado un especialista en inteligencia artificial y conocido, entre otras cosas, por sus predicciones basadas en su "Ley de los crecimientos acelerados", en la que de forma resumida, se explica que todo lo que nos rodea cambia de una forma más rápida de los que nos esperamos y esto, aplicado al mundo de los ordenadores, significa que los dispositivos son cada vez más potentes, cada vez mas pequeños y su evolución sigue su curso. De esta manera, predice que en la década del 2020 seremos capaces de introducir nanobots (robots del tamaño de un glóbulo rojo) en nuestro organismo con capacidad suficiente como para detectar las enfermedades y combatirlas desde dentro. Además postula que en el 2030 habremos descubierto cuál es el comportamiento de las neuronas humanas y éstas pasarán a ser una tecnología de la información, de forma que podremos imitar su comportamiento con un programa e introducirlas en nuestro cerebro haciéndonos más inteligentes. Por último en su obra predice que para la década del 2040 seremos capaces de descargar nuestros pensamientos y recuerdos en un ordenador, de manera que incluso podría resultar difícil distinguir el humano de la máquina.

A pesar de que las situaciones comentadas parecen muy futuristas, ya se están realizando experimentos dirigidos hacia estos objetivos y que dan una idea del camino que queda por recorrer.

La Inteligencia Artificial agrupa gran cantidad de técnicas de muy distinta naturaleza y cada una de ellas se debe utilizar según el ámbito o características del problema concreto. Para el escenario al que se enfrenta este proyecto se decidió que la más indicada era la de la Computación Evolutiva, dentro de la cual encontramos los Algoritmos Genéticos.

Entre las técnicas clásicas de la Inteligencia Artificial se encuentran las siguientes:

- Sistemas Expertos ([Harmon and King, 1985]). Los sistemas expertos basan su funcionamiento en la información proporcionada por un experto en el dominio en el que son aplicados. Se componen de una base de conocimiento (un banco de datos) y de un motor de inferencia, que consiste en una serie de reglas (normalmente if-else) que establecen la salida correspondiente a unas entradas, que normalmente son solicitadas a través de un formulario o encuesta. Los expertos proporcionan la información que luego procesará el motor de inferencia junto con los datos de entrada para ayudar a resolver un determinado problema. Una aplicación típica de estos sistemas son en aplicaciones de ayuda

a usuarios, que mediante un conjunto de preguntas cuyas respuestas generan una posible solución para un problema concreto. La aportación que un músico podría hacer al sistema presentado sería de gran utilidad, debido a la gran cantidad de conceptos a tener en cuenta, tecnicismos y variedad de posibilidades que ofrece este campo ya sea para medir el tempo, el compás, etc... Sin embargo el difícil acceso a expertos en esta materia ha hecho que en este caso no se haya podido contar con su ayuda.

- Redes de Neuronas Artificiales (RNA) ([Rosenblatt, 1958]). Son conjuntos de nodos lógicos interconectados entre sí de una manera compleja formando varias capas, y que simulan el comportamiento de las neuronas naturales. Las neuronas tienen pesos asociados. Haciendo uso de una determinada función calculan cuál es la salida correspondiente a partir de una entrada y su peso. Cada salida de una neurona de una determinada capa se conecta con las neuronas de la capa siguiente (convirtiéndose en sus entradas), de manera que la información se propaga a través de la red. El aprendizaje se basa en el método de prueba y error, es decir, la capa se somete a un entrenamiento, de forma que para unas entradas dadas y una salida esperada, se calcula el error cometido y en función del valor de este error se modificará la estructura de la red neuronal. Se puede decir por tanto, que la estructura es adaptativa en función del aprendizaje. Una vez entrenada la red, se somete a una serie de test (con datos aún no conocidos) para comprobar si la red ha sido bien entrenada. Dado que las RNA modelan relaciones complejas entre datos de entrada y salida y que pueden encontrar patrones en los datos, esta técnica es muy utilizada en problemas de predicción y clasificación, y en general en problemas en los que no se tiene un modelo general al inicio, pero sí una serie de ejemplos, como puede ser la predicción de series temporales. Por estas razones queda prácticamente descartado su uso para la creación musical, ya que lo que se pretende es obtener diferentes resultados y no una obra prestablecida. Es decir, el carácter dinámico del entorno musical hace que las RNA no sean una solución adecuada para ello.

- Agentes Inteligentes ([Wooldridge and Jennings, 1995]). Se trata de módulos software o hardware capaces resolver determinados problemas a base de técnicas de autoaprendizaje por medio de la experiencia, de forma autónoma y relacionándose con el entorno en el que son ejecutados. Partiendo de las percepciones y teniendo en cuenta el posible

estado interno, el estado del ambiente y los objetivos, el agente decide entre una de las posibles acciones que es capaz de llevar a cabo. El agente se vale de su propia experiencia, de los acontecimientos pasados, de sus propias capacidades, de probabilidades en los efectos de sus acciones, etc para llevar a cabo la tarea de elegir una determinada acción en un determinado momento. Hay tres tipos de agentes según la filosofía que éstos apliquen a la hora de tomar decisiones:

- Agentes Reactivos, en las que la decisión depende sólo del estado actual y de la percepción del agente acerca de éste.
- Agentes Deliberativos, en los que el agente tiene un modelo del mundo, un estado interno complejo, unos objetivos y es incluso capaz de predecir el resultado de sus acciones y planificar en consecuencia.
- Agentes Híbridos, los cuáles aplican una u otra de las dos filosofías anteriores según se cumplan determinadas condiciones.

Esta es una técnica que suele aplicarse en juegos y problemas en los que el entorno se encuentra predefinido y existe la posibilidad de aprender de pasos anteriores. Sin embargo, en el ámbito de la composición musical se hace complicado su uso debido a que el aprendizaje para alcanzar la solución óptima no es una opción viable al no existir la posibilidad de saber cuando un paso previo es mejor o peor que otro.

- Lógica Difusa ([Zadeh, 1965]). En los predicados de la lógica clásica no se tienen en cuenta ciertos factores que pueden aparecer en el mundo real. Existen problemas donde hay falta de datos, estos son incorrectos e imprecisos o se dan distintos resultados para el mismo problema. Estos hechos hacen que se tengan que deducir de otro modo. Aquí es donde entra la lógica difusa, que permite representar de forma matemática conceptos o conjuntos imprecisos. Para ello, parte de la idea de que los predicados no tienen por qué ser ciertos o falsos, sino que añade un grado de certeza a dichos predicados. Aparecen así calificadores como "un poco", "mucho", "algo", "bastante", los cuáles tienen un valor matemático asociado para que, a partir de unas premisas, se concluya o no un resultado. Añade además una probabilidad de que los resultados esperados se cumplan, lo que implica que la utilización de esta lógica en procesos de computación no permita generar resultados para los que se sepa de antemano que dadas unas entradas sean siempre fijos. Su uso en el caso que nos ocupa, aunque pueda parecer

aplicable debido al carácter cambiante del mismo y a la existencia de valores intermedios, no es lo ideal ya que el acto de componer no se trata tanto de una decisión en la que exista incertidumbre. Por ello, no se considera la opción más adecuada para afrontar este problema.

- Aprendizaje Automático. Las técnicas de aprendizaje automático se encargan de extraer reglas y patrones de grandes volúmenes de datos. Estos datos, antes de ser procesados, son tratados y computados en lo que se conoce como Minería de Datos o Data Mining ([Lovell, 1983]). Se trata de una técnica mediante la cuál los datos de entrada son preprocesados antes de aplicarlos a un proceso concreto. Tras ello, los datos que se presentan como entrada a un programa contendrán la mínima pero a la vez esencial información para resolver una determinada tarea. Los algoritmos de aprendizaje automático pueden ser de distinta naturaleza, desde RNAs hasta árboles de decisión basados en reglas sencillas (if-else). Dada la naturaleza del problema al que nos enfrentamos y la versatilidad de este tipo de técnicas parece que las condiciones son favorables para su aplicación y se podría estudiar su utilización con cierta confianza en la obtención de resultados satisfactorios o cuanto menos interesantes. Es una línea de desarrollo interesante que plantea un punto de vista diferente y atractivo a tener en cuenta. En el campo de la música existen ya algunos trabajos desarrollados, pero aplicados más a clasificación que a composición.

- Sistemas Híbridos ([Grossman et al., 1993]). Son sistemas lógicos independientes que hacen uso de dos o más técnicas de Inteligencia Artificial diferentes cada una de las cuáles desempeña un papel determinado para la consecución de los objetivos del sistema total. Por ejemplo, un uso extendido es el utilizar Algoritmos Genéticos como método de búsqueda de la mejor estructura de una RNA. Otro tipo de estos sistemas son los sistemas multiagente. En ellos se hace uso de varios tipos de agentes que realizan pequeñas tareas con el fin de lograr un objetivo común de todo el sistema. Cada uno de los agentes encapsula una determinada técnica según la subtarea que tenga que realizar y habitualmente hay un agente que se encarga de la comunicación entre el resto. El diseño y aplicación de este tipo de sistemas a la composición musical es una tarea compleja a la par que hasta cierto punto innecesaria. Sería interesante realizar ciertos experimentos orientados a obtener diversos resultados en función de los objetivos de cada integrante del sistema, pero para este caso no es aplicable debido a que,

en la mayoría de los pasos, el objetivo a alcanzar es el mismo y cada una de las partes deben trabajar orientadas hacia la misma dirección.

- Computación Evolutiva ([Holland, 1975] y [Fogel, 1994]). Las técnicas de computación evolutiva exploran un espacio amplísimo de soluciones para un problema dado pudiendo obtener de esta forma la mejor solución a dicho problema. Se basa en las mismas ideas que propuso Darwin en su artículo "El origen de las especies" ([Darwin, 1859]). Como es de sobra conocido, la teoría de la evolución de Darwin esta basada en tres principios fundamentales ([Griffiths et al., 2002]), que son el Principio de variación, el Principio de herencia y el Principio de selección.

  Darwin propuso una nueva explicación sobre el fenómeno aceptado de la evolución. Argumentó que la población de una especie en un momento determinado incluye individuos con características que varían de unos a otros. La población de la siguiente generación presentará una frecuencia mayor de aquellos tipos que sobreviven y se reproducen con más éxito, y siempre bajo las condiciones ambientales existentes. Estos conceptos son llevados a la informática y los procesos sometidos a este tipo de técnicas experimentan los mismos principios expuestos por el biólogo. Por ello, aplicando estas ideas, se puede utilizar la elevada capacidad de procesamiento de los ordenadores para, dada una población de individuos primitivos, poco inteligentes y obtenidos casi aleatoriamente, obtener una población de individuos final capaces de resolver un problema de la forma más eficiente posible. Para ello se utilizan operadores de selección, reproducción (cruce de individuos), mutación y evaluación de individuos (fitness) para ir obteniendo, a lo largo de la ejecución del algoritmo (varias generaciones de individuos), numerosas soluciones para un problema (cada una de las cuales se corresponderá con un individuo), de forma que sean las mejores soluciones las que permanezcan a lo largo de las generaciones.

  Tras estudiar las diferentes opciones comentadas y las implicaciones de cada una, se eligió utilizar computación evolutiva, y más concretamente Algoritmos Genéticos, debido a su elevada capacidad para buscar soluciones en espacios muy grandes y por la eficiencia probada en este tipo de problemas. Además a todo esto, hay que añadirle el valor experimental del enfoque, ya que, aunque las incursiones de la Computación Evolutiva en la música son una realidad, existen pocos trabajos que sigan esta línea y en aquellos que lo hacen, los resulta-

dos son aún susceptibles de mejora, con lo que no hay una idea clara acerca de como afrontar el problema. A continuación se presenta una explicación más detallada de las técnicas empleadas para el desarrollo del sistema.

1.1.1. Algoritmos Genéticos

Un Algoritmo Genético es en esencia un proceso de optimización genérico para obtener la mejor solución a un problema dado. Dicho problema puede ser de distinta naturaleza, por ejemplo, de clasificación (obtener el mejor clasificador), de diseño (obtener la mejor estructura de una red neuronal), etc. Este algoritmo realiza una exploración del mayor espacio de soluciones posibles, generando aleatoriamente una población inicial de individuos y sometiendo a dichos individuos a procesos de selección, cruce, mutación e inversión ([Mitchell, 1996]).

Un individuo encapsula una determinada solución para un problema, y está compuesto de distintos cromosomas o genes, cada uno de los cuales tendrá una función o rol específico para la resolución de dicho problema. Dado que no todos los genes de los individuos son los deseados, se utilizan los llamados operadores de cruce mediante los cuáles dichos genes son divididos e insertados en un individuo de la siguiente generación. Posteriormente se utilizan operadores de mutación mediante los cuáles los genes son modificados. Los individuos en cada generación son evaluados según una función específica (la llamada función de fitness, que es desarrollada para cada problema), y en base a dicha evaluación se seleccionaran y cruzarán los mejores individuos para formar la siguiente generación. De esta forma los genes más valiosos se mantendrán a lo largo de las generaciones, perdiéndose el resto de ellos, que no están lo suficientemente adaptados a la situación. Existen además otras configuraciones que se pueden utilizar con este tipo de algoritmos, como pueden ser el modelo de islas ([Darwin, 1845]) o el elitismo. La primera de ellas evita el estancamiento de individuos. Esto se consigue lanzando varias pruebas a la vez y llegado un momento determinado, un individuo de una de las ejecuciones es migrado a otra ejecución introduciendo sus genes en ésta. De esta manera puede variar la forma de evolución de esa población obteniendo mejores resultados y evitando los problemas ya mencionados. La segunda de las técnicas comentadas, el elitismo, permite conservar el mejor individuo de la población existente a lo largo de las generaciones. El modo de hacerlo es introduciéndolo directamente desde una generación a la siguiente sin pasar por las fases de selección, cruce y mutación. De esta manera siempre se asegura que el mejor individuo de la

población pase sus genes a la siguiente generación de individuos.

El uso de este tipo de técnicas es de gran utilidad en diversos ámbitos, entre ellos el caso que nos ocupa. Es posible modelar cada individuo de forma que sea una obra musical y se resuelva el problema de que las diferentes pistas e instrumentos suenen de un modo adecuado de manera conjunta. Los caminos por los que se pueden alcanzar estos objetivos son muy variados, por ello se desarrollarán varias funciones de fitness que nos permitirán comparar y mejorar los individuos, y por tanto las obras musicales generadas, iteración tras iteración. El mejor individuo vendrá dado en primer lugar por la repetición de ritmos o patrones siguiendo un determinado compás. Posteriormente se le irán añadiendo criterios y realizando mejoras a la función de fitness con el fin de obtener cada vez composiciones más elaboradas y de mayor calidad. Se mostrarán los resultados de los experimentos a cada paso de forma que se pueda comprobar la evolución del sistema a medida que se modifica la configuración del mismo.

Capítulo 2

Estado del Arte

A continuación se presenta una revisión del estado del arte en el área de la música y la inteligencia artificial. Se pretende dar una visión global y amplia sobre la literatura existente conocida por el autor, mostrando los distintos acercamientos realizados para afrontar este tipo de problemas, los avances que han supuesto cada una de ellas y los resultados más interesantes derivados de los experimentos.

2.1. Genetic algorithms and computer-assisted music composition (1991)

Diferentes enfoques se pueden realizar a la hora de emplear Algoritmos Genéticos para la composición musical. En este caso se propone un sistema mediante el cual se utiliza el algoritmo genético para seleccionar determinados operadores que se utilizarán para el proceso de composición ([Horner and Goldberg, 1991]).

El autor utiliza en su aproximación lo que ha llamado la "temática puente" (thematic bridging). Esto a grandes rasgos consiste en un procesos que permite transformar un patrón musical inicial en otro patrón diferente de una determinada duración. Esa transformación del patrón inicial se realiza a través de operaciones de modificación y reordenamiento sobre los elementos que forman el patrón inicial. De esta manera, la salida resultante será la concatenación de la salida resultante de cada una de las modificaciones realizadas. Un dato a tener en cuenta es que el patrón inicial no formará parte de la salida por sí mismo.

La siguiente tabla muestra de una forma más clara el funcionamiento de este sistema. Se parte de un patrón inicial de notas (Gb, Bb, F Ab, Db),

un patrón final a lograr (F, Ab, Eb) y de una duración de 17, con lo que el puente quedaría como sigue:

Operación	Patrón resultante
Borrar última nota del patrón inicial	Gb Bb F Ab
Rotación	Bb F Ab Gb
Borrar última nota	Bb F Ab
Mutar primera nota	Eb F Ab
Rotación	F Ab Eb

Cuadro 2.1: Proceso de transformación seguido por un patrón

La salida generada sería por tanto: Gb, Bb, F, Ab, Bb, F, Ab, Gb, Bb, F, Ab, Eb, F, Ab, F, Ab, Eb.

Para el proceso comentado anteriormente se debe de tener en cuenta que determinados conjuntos de operaciones puede que no permitan llegar del patrón inicial al final o que no se puedan ajustar a la longitud predeterminado. Para ello se comprueba, una vez ejecutado el algoritmo genético, si el resultado es aceptable o no. En caso negativo se modifica el patrón inicial, la duración o el conjunto de operaciones y se procede desde el principio hasta que se obtengan resultados deseables.

Los operadores que se pueden emplear son los siguientes:

- No-operation: No se realiza ninguna acción
- Añadir: Se añade un elemento al patrón
- Eliminar: Se borra un elemento del patrón
- Cambiar: Se cambia un elemento del patrón por otro
- Rotar: Se rotan los elementos que componen el patrón
- Intercambiar: Se coloca el elemento seleccionado en la posición dada por el segundo parámetro
- Rotar incorrecto: Rota los elementos colocados erróneamente con respecto al patrón final
- Intercambiar incorrecto: Intercambia elementos incorrectamente posicionados en el patrón

El cromosoma codifica estas operaciones de izquierda a derecha, de forma que cada operación se corresponde con un gen. Algunas de las operaciones mencionadas arriba pueden ir acompañadas de otros parámetros que pueden indicar, por ejemplo, la posición en la que se debe intercambiar un elemento. Cada parámetro se representa con un número real entre 0.0 y 1.0. Su valor será obtenido aleatoriamente. Además si ocurre la mutación otro número será elegido en sustitución de ese.

En cuanto a la función de fitness consiste en dos partes diferenciadas:

1. Se basa en lo próximo que es el resultado obtenido al patrón final que se está buscando. La fórmula empleada es:

   $$\text{Fitness} = 0.5 * (\text{Pcomún} + \text{Pcorrecto})$$

   donde el primer parámetro indica la proporción de elementos del patrón resultante que aparecen en el patrón deseado, y el segundo es la proporción de elementos del resultado que coinciden con elementos del patrón buscado.

2. Una vez que el patrón deseado y el resultante son coincidentes (fitness=1.0), es aumentado a través del cálculo de la duración del patrón resultante y el deseado, siendo fitness máximo cuando ambos coincidan. La evaluación de la segunda parte del fitness se realiza mediante la siguiente fórmula:

   $$\text{Fitness} = 1.0 + \text{Nd} - \text{abs}(\text{Nd} - \text{Nr})$$

   donde el primero es la duración deseada y el segundo la duración del patrón resultante. Aclarar que, en la práctica, duraciones algo superiores a las deseadas también pueden generar resultados aceptables.

Como se puede comprobar esta aproximación presenta un enfoque original para resolver el problema de la composición. El hecho de utilizar operaciones previamente definidas limita en ciertos aspectos el desarrollo del sistema pero por otro lado le confiere una serie de posibilidades que sería muy complicado alcanzar sin definirlas. Éste es uno de los primeros trabajos realizados en este campo y las ideas planteadas son muy interesantes aunque el propio autor deja claras líneas futuras y mejoras en la función de fitness que permitan mejorar la calidad de la música generada. Aún así, el empleo de patrones y su modificación ofrecen una buena oportunidad para generar

música de manera sencilla aunque con el "handicap" de que siempre debe existir supervisión humana debido a que los patrones deseados son elegidos en función de los gustos del usuario.

2.2. NEUROGEN, musical composition using genetic algorithms and cooperating neural networks (1991)

A continuación se presenta un sistema que combina algoritmos genéticos y redes de neuronas para alcanzar composiciones musicales con cierto orden y calidad ([Gibson and Byrne, 1991]). Para ello, la idea general del sistema es la de aplicar una RNA a ejemplos proporcionados por el usuario como buenos o malos de manera que se extraiga el conocimiento necesario y tras eso, aplicar el algoritmo genético que permita obtener modificaciones y configuraciones nuevas y beneficiosas. De esta manera la función de fitness del algoritmo genético vendría dada por la salida proporcionada por la red de neuronas.

Para poder alcanzar los objetivos, el autor decide simplificar el espacio de soluciones centrándose en un estilo particular que utilice un tipo de composición claramente definido, limitando la paleta musical disponible. Ese estilo será el de la armonía de cuatro bloques, es decir, utilizar cuatro partes (que podrían corresponderse con pistas) cada una de las cuales reproduce una melodía diferente y que se juntan para formar acordes. Esta elección viene dada por la libertad que permite y porque no requiere de un gran conocimiento semántico. Además sólo se considera un pequeño conjunto de notas en clave de C mayor (empleando sonidos en formato MIDI).

A la hora de afrontar el problema, el autor lo descompone en varios bloques: ritmo, melodía y armonía. Por lo tanto, la secuencia a seguir será la de construir el ritmo en primera instancia para después obtener la melodía dependiente de ese ritmo y por último lograr la armonía. Esta division simplifica el problema y aumenta las posibilidades de encontrar un mecanismo que alcance los requisitos propuestos.

La idea inicial del sistema era la de presentar una red neuronal con "buenos" y "malos" ejemplos y a partir de ahí encontrar una forma de generar "buenas" composiciones. A pesar de los intentos por llevar esta idea a una estructura de red neuronal, resultó difícil encontrar soluciones adecuadas para el dominio en cuestión. Se concluyó por tanto que las redes neuronales son creativamente estériles en este sentido y no podían generar nuevos estados del dominio. A partir de ese momento se intento aplicar un

algoritmo genético junto con la red de neuronas para, de esa forma, poder realizar una búsqueda paralela que permitiese encontrar la composición ideal basándose en las asignaciones como "buena" o "mala" de la red. De esta manera se aplicaba un algoritmo genético para cada uno de los bloques mencionados anteriormente guiados por la evaluación realizada por la red de neuronas para cada uno de esos bloques.

Posteriormente, se realiza la representación del sistema. Los cromosomas están formados por cadenas binarias en las que "1" representa un sonido y "0" ausencia de él. De esta manera se modelan cada una de las cuatro pistas que se utilizan en el sistema y también las que serán presentadas a la red para que, como el autor comenta, eventualmente reconozca las características que lo hagan bueno o malo.

De esta manera, los pasos que sigue el sistema son los siguientes:

1. Se modelan los cromosomas correspondientes a las pistas de la composición.

2. Se pasan a la red neuronal buenos y malos ejemplos con el objetivo de aprender las características deseadas para el ritmo.

3. Cuando la red ha aprendido, se pueden usar los resultados para guiar al algoritmo genético. Dominios pequeños permiten presentar a la red todas las posibles combinaciones de los ritmos y propagarlos para generar los correspondientes valores de fitness de cada uno.

4. El algoritmo genético comienza a trabajar empleando sus operadores:

   - Creación de ritmos posibles de manera aleatoria (población inicial).
   - Selección de dos individuos de la población utilizando el método Monte Carlo.
   - El cruce selecciona un bit de manera aleatoria y cruza ese bit de los individuos seleccionados anteriormente.
   - La mutación añade nueva información genética de manera aleatoria para prevenir la saturación.

5. Una vez obtenidos una serie de ritmos, se pasa al siguiente módulo para construir la melodía. Para ello se observa la relación y la estructura que existe entre las pistas mediante lo aprendido por a red de neuronas para este bloque, y se vuelve a ejecutar el algoritmo genético como en el caso anterior.

6. Se aplica la armonía a la melodía seleccionada. Éste es un proceso simple que consiste en la aplicación de varias reglas sencillas, asignando las notas correspondientes a cada elementos de la melodía.

El siguiente esquema representa los diferentes bloques y módulos por los que pasa la composición antes de alcanzar su versión final:

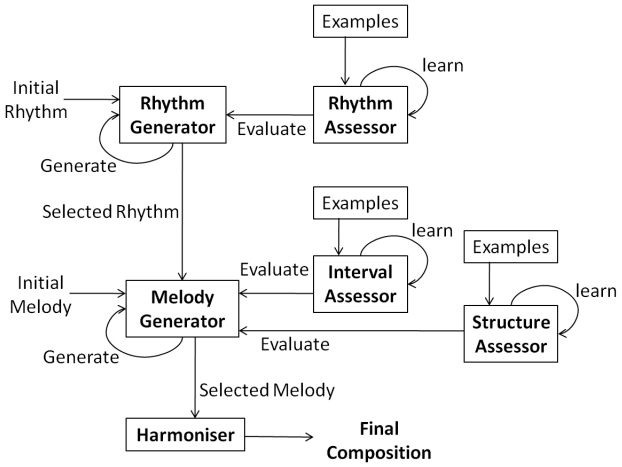

Figura 2.1: Proceso de composición de Neurogen

Tras observar varios experimentos se ha podido extraer información interesante derivada del funcionamiento del sistema:

- El hecho de que la mutación y el cruce dependan de ciertas probabilidades, puede afectar seriamente al resultado final.
- Las pruebas han demostrado que una alta probabilidad de cruce y una baja de mutación, mejora el rendimiento.
- La tasa de aprendizaje de la red y otros factores pueden afectar a la función de evaluación. La selección de estos parámetros sólo se puede determinar mediante ensayo y error.
- Los resultados parecen ser alentadores. Además, a partir de diferentes puntos en el espacio se producen resultados similares, lo que lleva a la conclusión de que el mecanismo es en realidad una búsqueda pseudo-original composiciones.

Con todas estas conclusiones se puede determinar que el sistema es efectivamente esperanzador, pero se debe de considerar el limitado ámbito en el que se está moviendo. Las redes neuronales han resultado ser moderadamente efectivas en pequeños dominios y el algoritmo genético permite la variación y generación de composiciones correctas pero muy reducidas y limitadas. Sin embargo, esto no se encuentra reñido con el interés generado, ya que la aproximación es realmente interesante y forma la base sobre la que seguir trabajando combinando las redes de neuronas y la computación evolutiva para obtener cada vez mejores resultados. Por otro lado, los requisitos hardware y software necesarios para este sistema son también muy escasos (se encuentra implementado usando C++, un ordenador personal, MIDI y un sintetizador multi-tímbrico) y pueden ser adquiridos sin problemas lo que lo hacen accesible y sencillo para cualquiera que se encuentre interesado en estas líneas de investigación.

2.3. GenJam: A Genetic Algorithm for Generating Jazz Solos (1994)

GenJam ([Biles, 1994]) es un sistema creado por John A. Biles, auténtico especialista en la aplicación de las tecnologías de la información a la música además de entusiasta de este arte y de la creatividad en general. Sus investigaciones en este campo, así como sus libros y artículos, son mundialmente conocidos y su nombre no pasa desapercibido entre los investigadores del sector. Tanto es así, que en este estado del arte se comentarán varias de sus obras debido a la gran relación de los contenidos tratados con los objetivos del proyecto.

Uno de sus logros más reconocidos en el que se centra esta sección, es la creación del sistema GenJam. A grandes rasgos y a través de la metáfora que él mismo utiliza, presenta un sistema en el que GenJam es un alumno de música que está aprendiendo a improvisar piezas de jazz. De esta manera, se encarga de reproducir piezas o conjuntos de notas recibiendo un feedback en tiempo real por parte de un mentor que otorga una determinada puntuación a la música generada por GenJam.

Como se puede observar, el planteamiento es sencillo, pero por el contrario, en el aspecto de diseño y desarrollo presenta varias particularidades que conviene comentar.

En primer lugar, el sistema esta desarrollado en base a un algoritmo genético cuya misión es aprender a improvisar música jazz, pero ese algoritmo no sigue exactamente los pasos de habituales. La primera diferencia

es la existencia de dos poblaciones en lugar de una. La primera de ellas es la que contiene la información acerca de los compases y la segunda las "frases" en sí (conjunto de sonidos generados). De esta manera un individuo en la población de compases coincide con una secuencia de eventos MIDI (tipo de sonido con el que trabaja) y un individuo en la población de frases está relacionado con los índices de los compases de la otra población. La otra gran diferencia con respecto al "estándar" es que GenJam usa las poblaciones completas para generar un solo, no una única medida o frase clasificada como la mejor. De esta manera estableciendo una concordancia entre compases y frases, se va generando un solo completo que será evaluado por el mentor.

La codificación de los cromosomas se hace de forma que cada uno de ellos está compuesto por cuatro números codificados en binario y a su vez cada uno de estos cuatro números son un puntero a la población de compases, estableciendo la relación comentada anteriormente.

Dos claras desventajas del sistema son que las notas sólo se suceden en múltiplos de 8 y sólo existen 14 tonos de los que sólo se puede elegir uno cada vez. Esto ciertamente supone una limitación considerable en la mayoría de los improvisadores humanos, pero permitiendo rítmica y una mayor diversidad cromática, aumentaría la longitud de la cadena de compases necesarios, ampliando el tamaño del espacio de búsqueda para la población de compases.

Una consideración interesante es que permite la ejecución de acordes mediante la reproducción simultánea de notas en la escala apropiada, que son reconocidas, procesadas y utilizadas en los solos. Esto provee de una formalización al sistema que le hace ser capaz de desarrollar ideas que pueden encajar en diferentes contextos armónicos y evitando además tocar la "nota equivocada".

En cuanto a la función de fitness, el autor decidió ser él la propia función de fitness, es decir, él se encargaría de evaluar cuando una improvisación es buena o mala. Este sistema hace que el evaluador humano sea un aspecto crítico del proceso y en algunos casos un cuello de botella o limitación más. Para desarrollar este sistema de evaluación, fueron necesarios varios puntos de ayuda al experto. Uno de ellos consiste en imprimir un pequeño retraso a la información que le llega al evaluador para poder realizar el feedback basándose también en la experiencia de composiciones pasadas.

En cuanto a los operadores, la selección se realiza por torneo modificado, en el cual se seleccionan 4 individuos aleatorios, de los cuales los dos mejores pasarán a ser padres y los dos peores serán reemplazados por la descendencia de esos padres seleccionados. De esta manera en cada generación la mitad de la población es sustituida, creando una proporción adecuada entre la

ganancia y la probabilidad de perder individuos necesarios para la correcta evolución del algoritmo. Para acelerar el proceso de aprendizaje, se introduce también la mutación. En este punto el autor hace especial hincapié debido a que utiliza seis tipos de mutaciones diferentes para poder de esa forma llevar a cabo sus objetivos.

Los resultados obtenidos por el sistema son satisfactorios a pesar de que se están aún realizando mejoras que permitan composiciones más elaboradas y adecuadas.

Como conclusión se puede decir que éste es un sistema que realiza un enfoque interesante en la aplicación de técnicas de computación evolutiva ya no sólo a la composición musical, si no a la generación de improvisaciones, con las características y dificultades que esto conlleva. Si al enfoque especial se le unen unas técnicas de desarrollo que rompen con lo que comúnmente se encuentra establecido en este campo, el valor añadido de este proyecto es considerable. El camino que abre es amplio y puede llevar a conclusiones más que reveladoras, además de servir de base para otros.

2.4. Generating rhythms with genetic algorithms (1994)

El siguiente sistema que se presenta es el creado por Damon Horowitz y consiste en la utilización de un algoritmo genético interactivo que aprende los criterios del usuario a la hora de generar ritmos musicales de manera que produzca aquellos que concuerden con sus propios gustos ([Horowitz, 1994]). De este modo el sistema desarrolla un modelo que representa los criterios del usuario y que se adapta a sus preferencias. La utilización de los algoritmos genéticos interactivos está claramente justificada, ya que de esta manera, el usuario puede dirigir la evolución del sistema sin necesidad de tener conocimientos musicales.

Debido a la gran cantidad de ritmos existentes, el autor decide examinar sólo un subconjunto de ellos de manera que pueda representarse la información musical necesaria como un simple vector simplificando el diseño. Para la representación del genotipo del ritmo se basa en los beneficios de usar una estructura genética diploide (doble notación de cromosomas) y tiene en cuenta aspectos como la creación de plantillas y la disposición y orden de esas plantillas a la hora de crear el fenotipo.

El proceso que sigue el sistema en la obtención de los ritmos pasa por, en primer lugar, la reproducción por parte del usuario de cada uno de los ritmos aleatorios de la población y la asignación de un fitness en función

del grado de satisfacción del propio usuario. A continuación se utilizan los operadores genéticos estándar de selección, cruce y mutación, y se procede de nuevo a evaluar. Para ello y para tratar con las dificultades propias de la subjetividad y variabilidad de estas tareas, existen también varias funciones objetivo a partir de las cuales el sistema puede evolucionar los ritmos:

- Síncopa
- Densidad
- Downbeat
- Repetición de pulsos
- Ritmo cruzado
- Funciones de clúster

Estas funciones son sólo unos ejemplos de los diferentes aspectos que se pueden incluir y pueden ayudar a la diferenciación de los ritmos además de ofrecer muchas más posibilidades creativas. El usuario podrá fijar el valor deseado de estos parámetros y su importancia relativa, lo que hará al sistema evolucionar de una manera consecuente a esos valores.

El sistema hace uso además de un algoritmo genético a nivel meta que se encarga de evolucionar la población de parámetros para que se acerquen a las funciones objetivo comentadas anteriormente. Para ello se adhiere a la técnica de los k vecinos más cercanos (k-nearest-neighbor) ([Punch, 1993]). De esta manera, evolucionando poblaciones de meta-individuos permite al usuario reducir rápidamente el espacio de búsqueda mediante la evaluación subjetiva de los ritmos generados por los meta-individuos sin tener que especificar los valores de las funciones objetivo.

Los resultados que se han recogido mediante la utilización de este sistema permiten obtener ritmos aceptables a partir de 50 evaluaciones realizadas por parte del usuario.

El sistema propuesto presenta ventajas importantes que lo hacen una herramienta muy recomendable para la generación de ritmos. Permite componer a usuarios sin conocimientos musicales y únicamente guiados por sus propios gustos. Además el amplio espacio de búsqueda en el que se puede mover hace que las composiciones puedan ser originales y variadas. Por otro lado, el proceso de composición puede resultar tedioso y, a pesar de las funciones objetivo creadas, existe la posibilidad de que al usuario le resulte complicado evaluar funciones, especialmente en los primeros pasos, en los

que le será difícil elegir entre ritmos generados aleatoriamente. Con sus pros y sus contras, lo que está claro es que el sistema supone una innovación a lo creado hasta el momento y las posibilidades que ofrece son muy numerosas. Las líneas futuras de este trabajo pueden incrementar aún más esas posibilidades y permitir obtener resultados competitivos con un menor número de evaluaciones, que es uno de los principales objetivos a conseguir.

2.5. Composing with Genetic Algorithms (1995)

Éste es uno de los artículos de referencia e interés en este campo. La base sobre la que se asienta el escrito de Bruce L. Jacob ([Jacob, 1995]) es la de combinar los métodos estocásticos con los sistemas basados en reglas complejas extrayendo lo mejor de cada uno, la simplicidad de los primeros con el determinismo de los segundos. Para ello, el autor explica como para este tipo de problemas, lo más recomendable es reducirlos a un problema de búsqueda. Al ser un espacio tan sumamente grande, se impone la aplicación de algoritmos genéticos de forma que se optimice la búsqueda de la solución más adecuada.

El objetivo principal que persigue este sistema es el de conseguir "composiciones que suenen bien" mediante variación de "frases" o conjuntos de notas. Como implementación para lograr esto, se presenta una sistema dividido en tres módulos:

- Composer: es el que produce la música
- Ear: es el encargado de filtrar el material no satisfactorio
- Arranger: se encarga de ordenar el material restante

Además, un operador humano juzga la calidad de los resultados anteriores que sirve para recombinar los individuos que han sobrevivido entre sí para obtener otros aún mejores.

Como se ha comentado, esta aproximación trabaja con conjuntos de notas o "motives" en lugar de con estructuras de bajo nivel, como serían las notas, para reducir el amplísimo espacio de búsqueda. Con lo que el sistema se encarga de construir pequeños "motives", evaluarlos y ordenarlos para construir otros de mayor longitud. Un dato importante es que para ese orden se impone que las notas de la pieza resultante estén relacionadas entre sí, para mantener de esa forma la cohesión de la composición.

El proceso de composición, por tanto, es el siguiente. El "composer" y el "ear" empiezan con unas primera configuración creada aleatoriamente que

necesitan ser modificados para adaptarse a los gustos del operador humano. Cuando se ha llegado a este punto, el operador humano define un conjunto de elementos que deben ser utilizados en la composición. El primer módulo se encarga de crear variaciones de esos elementos, usándolos para construir frases con uno de esos elementos cada vez. Al introducirse uno de esos elementos a las frases, el segundo módulo es consultado. Si éste no aprueba el resultado armónico del contenido, ese elemento resultante es eliminado. Cuando existen suficientes frases adecuadas, el tercer módulo se encarga de ordenarlas, evaluarlas y recombinarlas para obtener órdenes mejores.

Como se puede comprobar, éste es un enfoque muy interesante, ya que en base a unos deseos del usuario, se crean pequeñas frases musicales que pasan por una primera evaluación para, si son aceptadas, pasar a ser ordenadas de distintas maneras hasta conseguir el orden óptimo de las mismas, convirtiéndose en una buena base sobre la que hacer surgir un mayor número de experimentos.

2.6. Neural Network Fitness Functions for a Musical IGA (1996)

John A. Biles toma como referencia su anterior y conocido artículo GenJam ([Biles, 1994]) para experimentar con el uso de redes neuronales aplicadas a la creación musical ([John A. Biles, 1996]). Más concretamente pretende hacer frente al mayor problema de este tipo de sistemas, el diseño de un algoritmo para el cálculo de la función de fitness. Como se comentó previamente en este estado del arte, GenJam utiliza la función de fitness de un mentor (el usuario), que viene dada en tiempo real y permite al sistema saber si la improvisación que se está realizando es buena o mala. Lo que se desea en este caso es atravesar el cuello de botella que representa este método y automatizarla con la ayuda de una red de neuronas, o lo que es lo mismo, pasar de un fitness interactivo a un fitness automático.

Tras poner al lector en situación y explicar las principales características de GenJam, el autor continúa comentando las principales desventajas de su sistema de evaluación. Entre ellas se encuentra que a pesar de ser un método realista, un improvisador recibe el feedback de una improvisación mediante la reacción del público u otros músicos, en ocasiones resulta inconsistente o incompleta introduciendo ruido en el valor del fitness. Otro problema es que al depender del mentor humano se depende entre otros factores de su concentración. A pesar de que esto pueda aparecer como un hecho de escasa relevancia, la concentración es básica para esta tarea y es muy normal

que los humanos la pierdan, especialmente en las primeras fases de la improvisación donde las secuencias reproducidas se encuentran caracterizadas por la aleatoriedad. Este hecho también es un fuerte condicionante debido a que, inmerso en esa situación, se puntuará como buena cualquier secuencia que sea mínimamente musical y desde ese momento la composición se moverá hacia ese sentido.

Para solventar estos problemas comienza a definir las bases sobre las que se asienta la red de neuronas. El primer objetivo a alcanzar fue el de construir una red que sea capaz de reconocer individuos claramente "no-musicales" en la población de compases, de manera que las primeras generaciones de esos individuos puedan evolucionar sin la interferencia de aquellos claramente perjudiciales. La estructura de la red es de N nodos de entrada, M nodos ocultos y K salidas. Normalmente para este primer caso sólo existirá una posible salida, bueno o malo, y K pasaría a valer 1. El valor de M debe ser elegido con cuidado ya que valores muy bajos o muy altos pueden hacer que el sistema no funcione correctamente. Este problema fue resuelto utilizando el método *cascade-correlation*([Fahlman and Lebiere, 1989]). Esta técnica comienza construyendo una red simple, sin nodos ocultos, que entrena la red tan bien como puede. A continuación añade un nodo oculto que calcula el error de la red anterior y vuelve a entrenar la red. Cuando el entrenamiento se completa de la mejor forma posible, el nodo de salida se entrena para conseguir la salida deseada empleando como entradas todos los nodos de la red. Éste es un proceso rápido y permite saber si una aproximación es razonable, y si lo es, dar además el número ideal de nodos ocultos.

Los resultados de los experimentos arrojaron que grandes cantidades de nodos ocultos no respondían bien. Incluso aquellos que lograban aprender correctamente el conjunto de entrenamiento fallaban estrepitosamente cuando eran utilizados con el conjunto de test, indicando claramente que se producía sobreaprendizaje.

Otra parte de la red que se estudia es la formación de la capa de entrada. Para esto se utilizaron dos aproximaciones:

- Mediante la combinación de varios parámetros de manera estadística como pueden ser el número de nuevas notas en un determinado compás o el tamaño del máximo intervalo, es decir, la diferencia numérica entre notas adyacentes.

- Basado en histogramas de intervalos y notas en los datos, tanto por sí solos como en combinación con los parámetros anteriores.

Ambas aproximaciones fallaron y no obtuvieron resultados apropiados.

Todo ello lleva a la conclusión de que los humanos perciben y escuchan la música de una forma tan compleja que no puede ser representado de manera estadística. Por esta razón se encuentran numerosas situaciones en que cromosomas muy similares obtienen un valor de fitness casi opuesto.

Como se concluye en el propio artículo, la automatización del fitness puede resultar muy complicada y requerir amplios conocimientos tanto del campo de la inteligencia artificial como de música. Los enfoques planteados se presentan muy interesantes y a pesar de que los resultados obtenidos son poco satisfactorios, estas técnicas podrían funcionar aunque requerirían grandes recursos computacionales. Como ya se comentó, uno de los principales aspectos positivos de GenJam era que desde un escaso conocimiento de la música se confiaba en el mentor para suplir esa carencia guiando el proceso de creación musical. Desde esa perspectiva, no debería sorprender que automatizar el mentor aparezca como el gran obstáculo a sortear. Aún así el futuro es esperanzador y la ampliación de diversas técnicas y un conocimiento más profundo pueden incrementar en gran medida los resultados extraídos de este estudio.

2.7. Automatic composition of music by means of Grammatical Evolution (2002)

Como bien es sabido, la composición musical es un área multidisciplinar de gran interés. Hasta ahora se han podido estudiar diferentes enfoques que intentan afrontar este tipo de problemas desde distintos ángulos. En este artículo se presenta una nueva perspectiva, en la que se unen las gramáticas formales y las técnicas de computación evolutiva para acercarse al objetivo de realizar composiciones automáticas cercanas a las que compondría un ser humano([de la Puente et al., 2002]).

Se puede definir la evolución gramatical como una variación de la programación genética que utiliza gramáticas formales para representar las poblaciones. En este caso las gramáticas formales que se utilizarán en el sistema son únicamente las libres de contexto.

Un elemento esencial para el desarrollo de la idea y del sistema es la utilización del procesador auxiliar AP440, especialmente diseñado para reproducir música. Éste será el encargado de interpretar como música cada cadena de caracteres que siga una determinada sintaxis impuesta por el sistema (que será la conocida notación BNF).

Una vez definidos los principales aspectos del entorno, el funcionamiento

del sistema es como sigue. En primer lugar se deben seleccionar y traducir a melodías entendibles por AP440 melodías o canciones conocidas de compositores humanos. De esta manera, se puedan extraer y estudiar diferentes aspectos de las mismas con el objetivo de conocer qué parámetros son reconocidos como buenos para guiar así la composición posterior. Finalmente, la evolución de la gramática será utilizada para generar melodías con las características anteriormente expuestas de forma automática. Para ello también es necesaria la definición de una función de fitness que estará basada en la distancia euclídea entre los parámetros objetivo extraídos de las composiciones humanas y los parámetros de las melodías candidatas.

A continuación se detallan los principales aspectos del Algoritmo Genético empleado:

- Población inicial: estará compuesta de 64 vectores de 32 números enteros aleatorios entre 0 y 255.
- Cruce: se utiliza un cruce en una única posición elegida al azar.
- Mutación: cuando se decide mutar un individuo, la probabilidad de mutar cada posición es de 0.25.
- Dilatación: operador añadido a los clásicos comentados anteriormente que une el genotipo con otra cadena de números enteros.
- Corte: operador añadido que se encarga de eliminar alguno de los enteros contenidos en los vectores.

El número de individuos descartados en cada generación es de 16, es decir, un 25 % del total de la población, que serán reemplazados por la reproducción de los 16 mejores individuos elegidos al azar formando parejas. Además de todo lo ya comentado, juegan un papel fundamental las probabilidades, ya que el sistema adapta las probabilidades de cada operador, como por ejemplo durante el cruce, en el que si los padres son el mismo individuo, la probabilidad de mutación pasa a ser de 0.95 cuando habitualmente es de 0.25.

Como se puede observar, la solución que se propone mediante la aplicación de gramáticas formales a la composición musical es cuanto menos interesante. Este tipo de técnicas empleadas junto con los algoritmos genéticos aseguran un alto rendimiento y permiten explorar líneas escasamente tocadas hasta la fecha. Los resultados ofrecidos por este sistema conllevan el problema de ser altamente dependientes de las melodías humanas introducidas como referencia y, aunque el espacio de búsqueda es muy amplio, las

composiciones generadas serán siempre similares a las existentes, restándole un importante grado de originalidad a la composición. El hecho de que sea necesario el empleo de la notación BNF y su necesidad de traducción puede restar eficiencia en algunos momentos. Aún así, como los propios autores relatan, queda mucho camino por hacer y las líneas futuras son numerosas lo que deja una puerta abierta al crecimiento cualitativo de este tipo sistemas.

2.8. Evolutionary Music Composer integrating Formal Grammar (2007)

El objetivo de este artículo es presentar una herramienta capaz de componer música de forma automática empleando gramáticas formales y las reglas asociadas a ellas ([Khalifa et al., 2007]). Dichas reglas permiten generar y analizar secuencias de símbolos que en este caso se corresponden con parámetros musicales y sus atributos. Tras la ejecución de las distintas fases del sistema, se generan cuatro composiciones que serán traducidas a GMN (Guido Music Notation) que posee una representación alternativa en MIDI (Musical Instrument Digital Interface).

El proceso de composición que sigue el sistema se compone de dos fases que se presentan a continuación:

1. **Fase 1**
   En esta primera fase se generan los patrones musicales. Están formados por cromosomas, cada uno de los cuales se compone de 16 genes, permitiendo un máximo de 16 notas por patrón de sonido, cada uno de los cuales está limitado a una duración de una negra. El resultado final de esta fase es la creación de la tabla de los 16 mejores patrones de sonido en la que cada fila representa el propio patrón y cada columna las diferentes notas existentes en él. Aunque todos los patrones que se generan se adaptan a una misma duración, podrían estar compuestos de una, dos, cuatro, seis u ocho notas, aunque se remarca que los resultados obtenidos con sólo una nota no proporcionan resultados adecuados.

   Una vez realizados los pasos previos, se procede a realizar la evaluación. Es en esta parte del sistema donde se aplica un conocimiento musical más exhaustivo ya que se basa principalmente en conceptos como la formación de acordes al reproducirse varias notas a la vez, comprobar que la progresión de acordes sea adecuada y de acuerdo a los cánones marcados, estudiar la posibilidad de inversión de los acor-

des establecidos, hecho que también asegura un resultado adecuado, etc. Con todo ello se establecen una serie de secuencias adecuadas y usuales (por estar estudiadas estadísticamente) y se forma la correspondiente gramática libre de contexto. Entre sus características principales se encuentra la posibilidad de tener tuplas formadas por tres notas cualesquiera pero con la condición de que sigan la misma regla: una nota inicial, una nota tres tonos por encima de la inicial en la escala diatónica de C mayor, y una nota 4 tonos por debajo de la anterior en la misma escala. Cada uno de estos patrones será evaluado nota por nota y aquellas combinaciones que no puedan ser representadas por la gramática serán rechazadas mediante la asignación de un valor de fitness muy inferior al resto.

Además de la evaluación por parte de la gramática, el sistema se encarga de realizar la evaluación por intervalos. Este fitness se encarga de estudiar los saltos que existen entre unas notas y otras, calificando como positivos o negativos en cada caso. Además, tras el estudio de numerosos casos, se han determinado la adecuación de determinados valores concretos de saltos para los cuales, y si estos se produjesen, se otorgará además un bonus al valor del fitness.

2. **Fase 2**
   En esta fase los patrones que forman la tabla proveniente del paso anterior se combinan para formar frases.

   Se han aplicado también dos tipos de evaluación para esta segunda parte. La primera se corresponde con el análisis de los intervalos al igual que la fase anterior, que en este caso se encarga de comprobar si los saltos que se producen entre las notas que unen dos patrones son aceptables. El segundo es la evaluación de ratios, que se basa en la idea de que una buena melodía contiene un ratio ideal de tipos de notas y cualquier desviación de ese ideal conlleva una penalización en el fitness. Por ello se distinguen tres tipos de notas: *Tonos centrales* que componen los acordes de una clave, *Notas de color* que son el resto de notas de una clave y *Notas cromáticas* que son todas las notas al margen de las anteriores. Con esto, la proporción de notas ideal que se propuso fue la de 60 %, 35 % y 5 % respectivamente. Estos porcentajes se presentan como un punto de partida y en futuros experimentos pueden ser modificados.

Se puede concluir que los resultados obtenidos mediante este sistema son adecuados y melódicamente correctos. Las composiciones resultantes siguen

un orden establecido, tal vez demasiado restringido, pero que hacen que el sonido generado se acerque un poco más a las composiciones humanas asegurando por tanto un mínimo de calidad. La forma en la que se presenta es adecuada, no parte de canciones ya existentes y trata de realizar composiciones originales, aunque tal vez el hecho de poseer reglas tan restrictivas pueda comprometer la originalidad que se persigue. Se hecha en falta en el artículo una mayor explicación del algoritmo genético empleado en la generación de los patrones al que sólo se hace referencia mencionando que opera sobre la población inicial de cromosomas.

2.9. Evolutionary Computer Music (2007)

Este libro de reciente edición recoge la información más completa e interesante relacionada con el empleo de técnicas de computación evolutiva aplicadas a la música ([Miranda et al., 2007]). Su estructura y la colaboración de varios de los autores más reconocidos de este área, lo convierten en un perfecto punto de partida, y además puede servir de referencia para quienes tengan como objetivo profundizar más aún en su conocimiento o buscar nuevas fuentes de inspiración para sus experimentos.

Los primeros capítulos sirven de toma de contacto para el lector, realizando una introducción a los principales conceptos relacionados con el funcionamiento de los algoritmos genéticos y sus posibles enfoques y tratamientos dentro del campo de la música. También se hace referencia a otros aspectos como la creación de estructuras en función del tiempo, modelización de poblaciones, operadores genéticos, tipos de fitness y cómo conseguirlos, todos aspectos a tener en cuenta al utilizar estas técnicas aplicadas a la música, así como problemas que se encontrarán y sus posibles soluciones, etc. También se reserva una serie de apartados a definir y comentar nociones puramente musicales, lo que resulta de gran ayuda para posteriores capítulos del libro y para poder comprender y desarrollar las ideas planteadas en su totalidad.

Posteriormente, y tras realizar un breve recorrido por la evolución y el diseño del audio digital, se presentan varias aplicaciones y sintetizadores que permiten una mayor facilidad de manejo de estos aspectos. Como ejemplo de ellas encontramos MutaSynth, un sistema que permite sintetizar sonidos con evolución interactiva ([Dahlstedt, 2001]).

A partir de ese momento el libro se centra en temas y enfoques mucho más concretos. Los diversos autores que participan exponen sus principales ideas, proyectos y sistemas explicando detalladamente cada una de sus

aproximaciones, experimentos y resultados. Entre los más interesantes se encuentra los siguientes:

- El ya comentado GenJam ([Biles, 1994])
- Experimentos realizados por Eduardo R. Miranda y Qijun Zhang que prueban el rendimiento de la creación musical mediante algoritmos evolutivos. En ellos tratan de dar respuesta o encontrar similitudes entre las diferentes desviaciones que tienen lugar en las piezas musicales, teniendo en cuenta incluso factores sociales que puedan influir en la percepción musical
- GenDash, un sistema creado por Rodney Waschka II y utilizado únicamente por él durante años. Este sistema es capaz de componer piezas que abarcan desde un único intérprete a cuartetos de cuerda o incluso piezas instrumentales para orquesta con las que consiguió resultados más que aceptables interpretándolas en diferentes países, grabando discos y consiguiendo la financiación correspondiente.

Como se ha podido observar, este libro es la prueba de que la evolución en la creación musical por ordenador, y más concretamente mediante la utilización de algoritmos genéticos, es una realidad. El interés que despierta este campo está en continuo aumento y, por ello, los avances y las cotas alcanzadas están cada vez más próximas al sueño de una composición musical automática de alta calidad. Este libro muestra además el gran potencial de la computación evolutiva y la habilidad de adaptarse a las diferentes situaciones que se puedan plantear. Los resultados, aunque mejores, se encuentran todavía lejos del ideal deseado, pero el gran surgimiento de ideas y la capacidad creciente de cada una de las posibilidades hacen que el futuro de la composición musical evolutiva sea prometedor.

2.10. Genetic Algorithms and the abc Music Notation Language for Rock Music Composition (2008)

El más reciente de los artículos que se comentan en este estado del arte presenta un sistema multi-instrumental capaz de componer música rock mediante la aplicación de algoritmos genéticos ([Oliwa, 2008]). Sin duda, este sistema representa un avance muy importante respecto a los comentados anteriormente y los resultados obtenidos son realmente interesantes.

El propio creador del sistema lo define como único debido a su enfoque y perspectiva, y verdaderamente lo es, no tanto por el propio algoritmo genético empleado o por la representación del mismo, sino porque, entre otras cosas, utiliza el lenguaje *abc*, un lenguaje de descripción musical en formato ASCII. Esta notación permite representar complejas estructuras musicales sin reducirlo a notas, duración de notas, acordes, etc. El sistema será capaz por tanto de computar los símbolos de este lenguaje codificados como enteros en los alelos del genoma, usar funciones de transformación que asignan un símbolo distinto a cada uno de esos enteros y convertir el mejor de los individuos a un fichero en este lenguaje al final del proceso.

Debido a que cualquier canción puede ser traducida desde MIDI al lenguaje abc, la salida generada por el sistema será un archivo MIDI.

¿Y cuál es el proceso que puede seguir un sistema para componer música rock? En primer lugar, definir y estudiar las principales características que poseen las composiciones de este estilo musical. Como se ha podido comprobar en los estudios anteriores, este tipo de sistemas suelen ceñirse a una única escala musical, sin embargo este sistema utiliza tres implementaciones diferentes de la escala de E menor, la más importante de la música rock:

- **Escala de Blues de E menor**: hace especial hincapié en notas que producen sentimiento de melancolía.
- **Escala Melódica de E menor**: la escala habitual de la música rock.
- **Escala Armónica de E menor**: elevada siete grados con respecto a la escala melódica que produce un sonido neoclásico.

El sistema es capaz de componer melodías para 4 instrumentos diferentes, aunque la cantidad y el tipo de ellos puede cambiar de unas composiciones a otras:

- **Guitarra principal**: representa un virtuoso del instrumento a través de técnicas especiales que se comentan más adelante.
- **Guitarra rítmica**: construye la base del ritmo a través de acordes y punteos.
- **Piano/Órgano**: amplia el espectro de posibilidades en cuanto al ritmo y la melodía.
- **Batería**: proporciona el ritmo de la música con sus golpes.

Una vez definidos las principales características musicales que abarca el sistema, es momento para explicar en profundidad los diferentes aspectos del algoritmo genético empleado. La representación de los individuos se realiza mediante el genotipo, que lo forma un array multidimensional de genes creado de forma arbitraria pero fija, cada uno de los cuales está asociado con un alelo de enteros con un rango limitado y mediante el fenotipo, que está formado por las notas de cada uno de los instrumentos usados en la composición.

La estructura interna de los individuos, está formada por un conjunto de genomas de k alelos (donde k representa el número de instrumentos que se han escogido previamente) que contienen números enteros. Existen 5 tipos de genomas:

1. Compuesto por el conjunto de alelos de números enteros para la guitarra principal con 25 o 29 valores de tonos de notas. Se incluyen 7 notas de la escala melódica o armónica de E menor. Las notas se presentan en cuatro octavas. También se incluye el operador de pausa que inserta una pausa en lugar de un sonido.

2. Compuesto por el conjunto de alelos de números enteros para la duración de las notas de la guitarra principal. Se utilizan 7 longitudes de nota diferentes.

3. Compuesto por el conjunto de alelos de números enteros para los acordes fuertes de la guitarra rítmica y posee 44 valores. Los valores corresponden a 11 acordes fuertes en la escala de E menor, cada una de las cuales puede tener 4 duraciones diferentes.

4. Compuesto por el conjunto de alelos de números enteros para los patrones de la batería con 15 valores. 12 se corresponden con patrones básicos de batería y 3 con las velocidades que se pueden aplicar a esos patrones.

5. Compuesto por el conjunto de alelos de números enteros para los acordes del piano o el órgano, que usa la misma representación que la guitarra rítmica.

Al inicio de la composición se produce una división de la canción en segmentos de manera aleatoria pero que se mantendrá fija a lo largo de la evolución de la misma. Cada uno de los segmentos tendrá por tanto una longitud arbitraria. Cuando se hayan fijado esos segmentos se realiza la asignación de un tipo de fitness concreto, del conjunto de fitness existentes

para cada genoma, a cada uno de esos segmentos. Esa asignación también se mantendrá a lo largo de toda la ejecución del algoritmo genético. La función de fitness otorgará tantos puntos a un determinado segmento como valores contenidos cumplan los requisitos establecidos. El fitness total vendrá dado de la suma de los fitness de cada segmento.

A continuación se comentan brevemente los diferentes tipos de fitness existentes para cada genoma:

1. Genoma 1 - Notas Guitarra Principal
    - **Ascender en la escala**. Esta función crea escalas ascendentes, es decir, cada tono debe ser superior al inmediatamente anterior a él. El individuo obtiene una recompensa por cada par de notas que cumplan esta norma.
    - **Descender en la escala**. Función contraria a la anterior. El individuo obtiene la recompensa por cada par de notas en las que la primera sea superior a la segunda.
    - **Tapping**. Se produce cuando hay una rápida alternancia entre una nota baja base y otras que son todas más altas o más bajas que la nota base.
    - **Suavizar**. Esta función limita el salto entre notas correlativas creando una melodía fluida apropiada para pasajes lentos.
2. Genoma 2 - Duración Notas Guitarra Principal
    - **Longitud de la nota**. Controla que la duración de una secuencia de notas sea apropiada y siga unos cánones establecidos.
3. Genoma 3 - Acordes Guitarra Rítmica
    - **Ascender mediante acordes**. Función que realiza la misma operación que la ascensión de la escala en la guitarra principal.
    - **Descenso mediante acordes**. Función análoga al descenso en la escala de la guitarra principal.
    - **Ritmo con énfasis en E**. Como el propio nombre indica, toma esa nota como referencia y realiza un elevado uso de la misma.
    - **Ritmo galopante**. Similar a la anterior en la que la nota base no tiene que ser E.
4. Genoma 4 - Patrones Batería

- **Bloques**. Función que controla que un patrón predefinido se repite un número de veces. También asegura la existencia de un número pequeño de pausas durante la composición.

5. Genoma 5 - Acordes Piano/Órgano

   - **Acordes**. Esta función se encuentra optimizada de la misma forma que los acordes para la guitarra rítmica. Existen restricciones para evitar que se ascienda en la escala y pueda en sentido contrario al de la guitarra.

En cuanto a los operadores genéticos que se han empleado, destaca la utilización de un cruce multipunto (2 puntos), una probabilidad de recombinación del 80 %, una mutación gaussiana con probabilidad de 0.1 y selección por torneo, todo ello con una población de 200 individuos. Con estos parámetros una canción completa se construye en 11 segundos y permite obtener medio minuto de música.

Como se ha podido comprobar, este es uno de los más complejos sistemas presentados, ya que el número de características y posibilidades que se tienen en cuenta es mucho mayor. Lo que en un principio pueden aparecer como demasiadas restricciones se tornan en opciones para el sistema, que en lugar de centrarse en obtener un único objetivo, divide la composición en diferentes partes que persiguen finalidades diferentes, dotando a la composición final de mayor originalidad y dinamismo, ya que también se tiene en cuenta que la unión de esas diferentes partes sigan una misma línea. Merece la pena destacar una vez más el uso de la notación abc como un paso intermedio, que permite combinar la potencia de los algoritmos genéticos con una rica representación musical. Los resultados obtenidos finalmente son cuanto menos curiosos ya que, aunque con una calidad reducida debido a los sonidos en formato MIDI, permiten identificar perfectamente el estilo musical que se está trabajando sin tomar como referencia otras piezas que pudiesen restarle originalidad. Sin duda el sistema presentado es único hasta el momento y parece que contribuirá a que el rock siga vivo.

Capítulo 3

Análisis del problema

3.1. La música digital

La música digital es un campo que ha ido cobrando cada vez mayor interés en el mundo de los computadores, lo que ha incrementado su alcance, evolución y posibilidades entre otras cosas por las numerosas ventajas que ofrece el tratamiento digital de señales (facilidad de transmisión, inmunidad al ruido, no se degrada con el tiempo, etc).

En sus inicios la música digital presentaba un formato muy enfocado al usuario. Su finalidad era únicamente la de hacerle llegar las melodías previamente creadas y convertidas a un formato legible y reproducible por el ordenador. Inmediatamente este suceso dejó entrever las grandes posibilidades que ofrecía el mundo de la música digital. Si éramos capaces de reproducir música en el ordenador, ¿porque no íbamos a ser capaces de generar música mediante él? Estas nuevas ideas y las tendencias musicales en continua evolución precipitaron las primeras incursiones de la informática en el mundo de la música.

Desde ese momento han ido surgiendo formatos, programas, lenguajes y organizaciones especialmente enfocados a este área tan interesante. La, como siempre, rápida evolución del mundo de la informática, ayuda enormemente al surgimiento de nuevas posibilidades, tanto hardware como software, que hacen mucho más sencillo, interesante y productivo el manejo de música mediante el ordenador. Por todos estos motivos, la creación de música digital ha llegado a ser en poco tiempo un recurso necesario en cualquier discográfica y ha provocado la existencia de grupos y artistas cuyo soporte musical está íntegramente creado por ordenador([Miranda, 2001]).

Ligado a estos conceptos, se encuentra el campo de la Inteligencia Ar-

tificial. Se pueden usar técnicas provenientes de este área para optimizar y mejorar el rendimiento de aplicaciones, sistemas y para resolución de problemas complejos. Si aplicamos y relacionamos estos principios con la creación musical, el gran interrogante que alcanzamos es, ¿Se podría mejorar la creación musical usando la Inteligencia Artificial? Y extendiendo nuestras miras un poco más lejos, ¿Se podrían componer obras musicales empleando solamente la Inteligencia Artificial?

Este capítulo se encargará de presentar y analizar las diferentes opciones y tecnologías existentes que mejor se adaptan para alcanzar los objetivos planteados. Antes de entrar en profundidad a comentar estos estudios, se fijarán los objetivos principales que debe cubrir este proyecto.

3.2. Objetivos

El presente proyecto tratará de resolver el problema de la composición semi-automática de música mediante técnicas de computación evolutiva. El principal objetivo es el de lograr composiciones que se ciñan a unos parámetros de configuración dados. Estos parámetros pueden ser la duración de la composición, el compás o los instrumentos a incluir. La salida que debe producir el sistema es una composición agradable al oído y con un cierto orden y estructura, es decir, que reproduzca una canción con un cierto ritmo y según los cánones musicales establecidos.

Se tratará por tanto de optimizar al máximo la calidad de la composición producida por el sistema, manteniendo unos niveles adecuados de tiempo y coste de ejecución. La relación con otro tipo de problemas, los resultados de los experimentos producidos y las diversas aplicaciones que puede tener esta aplicación, confieren gran interés al proyecto.

Además de los objetivos propios del proyecto se buscará una implementación adecuada de forma que se minimice el impacto producido por cualquier modificación posterior en el análisis o diseño.

Con todo ello se pretende que las composiciones generadas por el sistema puedan servir de base para generar melodías más complejas sobre ella, de fondos rítmicos para usos diversos e incluso de fuente de inspiración, que sirva de punto de partida para músicos de forma que les ayude a crear música similar a partir de la obtenida como salida del sistema.

Previamente al desarrollo de la aplicación, se realizó un primer estudió de viabilidad del proyecto. Se desarrolló una API para el manejo de sonido para asegurar el correcto tratamiento del mismo y que las tecnologías existentes permitían llevar a cabo los objetivos del proyecto. Los resultados obteni-

dos en las primeras aproximaciones fueron satisfactorios y permitieron la continuidad del proyecto.

Una vez alcanzado dicho punto, se pasó a un estudio más profundo de la situación actual y de las posibilidades que se ofrecían en este ámbito. El fin de estos estudios era el de alcanzar el mayor grado de conocimiento posible del entorno y sus características, limitaciones y requisitos para poder construir el sistema de la forma más óptima y adecuada, tanto desde el punto de vista tecnológico como de rendimiento.

A continuación se detallan los diferentes estudios realizados así como las conclusiones más importantes extraídas de los mismos.

3.3. Conceptos necesarios para el manejo de la música digital

La música digital es un concepto bastante moderno que desde sus inicios hasta ahora ha sufrido multitud de cambios tanto en la manera de ser concebida como en las formas y formatos en los que se presenta. Relacionado con esto, existen varios términos que deben ser explicados para poder entender completamente el ámbito en el que se encuentran([Vignoli, 2004]):

3.3.1. Musical Instrument Digital Interface (MIDI)

A principios de los años 80, la tecnología de los sintetizadores había conseguido avances importantes en su empeño para conseguir instrumentos capaces de reproducir sonidos espectaculares tanto, creando imágenes de sus correspondientes acústicos, como produciendo tonos irreales inventados por la imaginación de algunos músicos, que encontraron en estos sonidos nuevas inspiraciones y medios para su creatividad. Sin embargo, uno de los problemas que permanecían sin solucionar era la incompatibilidad entre diferentes instrumentos.

Si se considera que los sintetizadores eran monofónicos, es decir, sólo eran capaces de producir una sola nota a un tiempo, es posible imaginar como estos aparatos eran incapaces de competir con los verdaderos pianos o las guitarras, que eran dos de los instrumentos más usados en la música moderna. Este problema conllevó a que en 1982, Dave Smith de la empresa fabricante Sequential, se propusiera conseguir el "milagro"de poner de acuerdo a las grandes compañías para crear un protocolo o norma de comunicación entre los instrumentos que fuese respetada por todos los dispositivos. La idea básica era permitir hacer sonar a más de un aparato a

la vez, creando así un instrumento polifónico por el sistema de adición de varios componentes.

Las especificaciones se prepararon a mediados de 1982 y se publicaron a finales del mismo año bajo el título « The Complete SCI MIDI» abreviándose finalmente a M.I.D.I. que es el acrónimo de "Musical Instruments Digital Interface"(interfaz digital para instrumentos musicales). El sistema es simple de instalar y su estructura sencilla, barata y efectiva.

Todo equipo que esté etiquetado como MIDI debe cumplir una serie de características tanto hardware como software que lo hagan compatible con el resto de dispositivos.

Los aspectos que se tuvieron en cuenta para desarrollar el protocolo MIDI fueron los siguientes:

- Tono o altura: parámetro que representa las vibraciones por segundo de la señal de audio. Determina lo agudo o grave que será el sonido. Normalmente se trabaja con una escala de 0 a 127, aunque no todos los dispositivos llegan a alcanzar estos niveles, en cuyo caso se ignoran o se desplazan a otra escala (Hertzios).

- Intensidad o volumen: grado de fuerza con el que se emite un sonido. Se mide en Decibelios (dB).

- Voz: elementos de generación de sonidos. Pueden ser notas para un sintetizador, fonemas para una alocución o elementos de percusión para una caja de ritmos.

- Timbre: diferenciación entre un sonido y otro.

- Pulsación o velocidad: fuerza con la que se pulsa, mantiene o suelta un sonido. Se suele medir en escala logarítmica. El 0 se considera el valor mínimo y 127 el máximo. En ausencia de este dato se considera el 64.

- Duración: indica el tiempo que dura un sonido.

- Canal MIDI: es el concepto más complicado de entender y explicar. Se basa en los siguientes principios:

  - MIDI especifica 16 para la transmisión de datos entre dispositivos, es decir, para direccionar los mensajes que se envían de un instrumento a otro.
  - El canal número 10 se utiliza para ritmos y baterías.

  - Los datos se pueden estar transmitiendo por todos los canales a la vez o por uno o varios canales individuales.
  - Los datos de un canal individual no tienen efecto alguno sobre los que se reciben por otro canal diferente, son independientes.

- Latencia: es el tiempo que transcurre desde que se introduce una nota hasta que se procesa y se reproduce.

3.3.2. La música en los ordenadores personales

Como se comentaba al inicio de la sección, el avance conjunto de los procesadores y las tarjetas de sonido, han propiciado que los nuevos programas utilicen una serie de ventajas para generar sonidos y componer música. Así, a la tecnología Wavetable (tabla de ondas para emular MIDI) se une la capacidad de un procesador actual de reproducir decenas de sonidos a la vez, lo que por fin ha puesto al alcance de cualquier persona el tratamiento prácticamente profesional de sonidos. La única dificultad será la de hacerse con bancos de sonido de calidad, que en ocasiones son realmente caros, o bien generar nuestros propios sonidos. Si lo que se desea es fabricar nuestro propio compositor, algunos lenguajes de programación como Java ofrecen la posibilidad de descargar un banco de sonidos interesante pero que puede resultar insuficiente si se persiguen resultados de una cierta calidad.

En este tema conviene comentar la existencia de ASIO, una tecnología multiplataforma desarrollada por Steinberg para la transferencia de audio multicanal. Sus principales ventajas son, entre otras, que permite aumentar las prestaciones de las tarjetas estándar de sonido, de forma que se aumenta el número de entradas y salidas disponibles de la tarjeta.

3.3.3. Almacenamiento de audio

El primer formato de almacenamiento de audio fueron los archivos de extensión ".wav", aunque posteriormente estos han ido viéndose reemplazados por archivos "mp3" que utilizan un sistema de compresión más avanzado. El principal problema de los archivos "wav" es su gran tamaño.

El tamaño de un archivo de audio viene definido principalmente por tres características:

1. Duración del sonido: es un factor dependiente de cada sonido.

2. Frecuencia de muestreo: se mide en Hertzios e indica la frecuencia con la que se toman los datos de la señal para almacenarlos y luego

reconstruir la señal. Si no se toman datos con la suficiente frecuencia la señal podría perderse, por otra parte, cuanto mayor sea la frecuencia, mayor será la precisión del sonido.

3. Calidad del muestreo: indica cuanto información tomar de cada medición, normalmente 8, 16, 24, etc. bits. Un mayor número aumenta la nitidez del sonido, pero también el tamaño del fichero.

Otros formatos de audio permiten comprimir el sonido hasta diez veces incluso sin pérdidas significativas de calidad, obteniendo una muy buena relación calidad/compresión.

WAV

WAV (o WAVE) es el apócope de WAVEform Audio Format. Es un formato de audio digital, normalmente sin compresión de datos, desarrollado y en propiedad de Microsoft y de IBM, que se utiliza para almacenar sonidos en el ordenador. Admite archivos mono y estéreo a diversas resoluciones y velocidades de muestreo y su extensión es .wav.

Es una variante del formato RIFF, método para almacenamiento en "paquetes", y relativamente parecido al formato AIFF usado por Macintosh.

A pesar de que el formato WAV puede soportar casi cualquier códec de audio, se utiliza principalmente con el formato PCM (no comprimido) y al no tener pérdida de calidad puede ser usado por profesionales. Para tener calidad disco compacto (CD) se necesita que el sonido se grabe a 44100 Hz y a 16 bits. Por cada minuto de grabación de sonido se consumen unos 10 megabytes de disco duro. Una de sus grandes limitaciones es que solo se puede grabar un archivo de hasta 4 gigabytes, que equivale aproximadamente a 6,6 horas en calidad disco compacto. Es una limitación propia del formato y se debe a que en la cabecera del fichero se indica la longitud del mismo con un número entero de 32 bit, lo que limita el tamaño del fichero a 4 GB.

En Internet no es popular, fundamentalmente porque los archivos sin compresión son muy grandes. Son más frecuentes los formatos comprimidos con pérdida, como el MP3.

MP3

MPEG-1 Audio Layer 3, más conocido como MP3, es un formato de audio digital comprimido con pérdida desarrollado por el Moving Picture Experts Group (MPEG) para formar parte de la versión 1 (y posteriormente ampliado en la versión 2) del formato de vídeo MPEG.

Este formato fue desarrollado principalmente por Karlheinz Brandenburg, director de tecnologías de medios electrónicos del Instituto Fraunhofer IIS que junto con Thomson Multimedia controla el grueso de las patentes relacionadas con el MP3. La primera de ellas fue registrada en 1986 y varias más en 1991. Pero no fue hasta julio de 1995 cuando Brandenburg usó por primera vez la extensión .mp3 para los archivos relacionados con el MP3 que guardaba en su ordenador. Un año después su instituto ingresaba en concepto de patentes 1,2 millones de euros. Diez años más tarde esta cantidad ha alcanzado los 26,1 millones.

El formato MP3 se convirtió en el estándar utilizado para streaming de audio y compresión de audio de alta calidad (con pérdida en equipos de alta fidelidad) gracias a la posibilidad de ajustar la calidad de la compresión, proporcional al tamaño por segundo (bitrate), y por tanto el tamaño final del archivo, que podía llegar a ocupar 12 e incluso 15 veces menos que el archivo original sin comprimir.

Fue el primer formato de compresión de audio popularizado gracias a Internet, ya que hizo posible el intercambio de ficheros musicales. Tras el desarrollo de reproductores autónomos, portátiles o integrados en cadenas musicales (estéreos), el formato MP3 llega más allá del mundo de la informática.

A principios de 2002 otros formatos de audio comprimido como Windows Media Audio y Ogg Vorbis empiezan a ser masivamente incluidos en programas, sistemas operativos y reproductores autónomos, lo que hizo prever que el MP3 fuera paulatinamente cayendo en desuso, en favor de otros formatos, como los mencionados, de mucha mejor calidad. Uno de los factores que influye en el declive del MP3 es que tiene patente. Técnicamente no significa que su calidad sea inferior ni superior, pero impide que la comunidad pueda seguir mejorándolo y puede obligar a pagar por la utilización de algún códec (lo que ocurre con los reproductores de MP3). Aún así, a inicios del 2008, el formato mp3 continua siendo el más usado y el que goza de más éxito.

3.3.4. Efectos

Un aspecto interesante de la música digital es la capacidad de modificar cualquier sonido al gusto del compositor. Esto se consigue mediante la introducción de efectos en los sonidos que pueden hacer que el sonido emitido por cualquier instrumento se distorsione para producir una sensación diferente.

Existen gran cantidad de efectos musicales. Por su naturaleza se puede establecer la siguiente clasificación:

- Basados en retardos: la base de este tipo de efectos es el retardo de

la señal en el tiempo. Ejemplos de este tipo pueden ser el eco o la reverberación.

- Basados en filtrado: como su propio nombre indica, surgen del filtrado de la señal original.
- Basados en la amplitud: consisten en la alteración de la amplitud de la señal y que pueden dividirse en efectos simples y sofisticados.

Basados en retardos

Los efectos que utilizan **retardos**, muchas veces se consiguen sumando a la señal original varias copias retardadas y modificadas de diversas formas. Según el tipo de efecto que se desee conseguir los tiempos de los retardos pueden variar desde las pocas milésimas a varios segundos. Como se comentó anteriormente, los mas típicos son el eco y la reverberación.

- **Reverberación:** es la suma total de las reflexiones del sonido que llegan al lugar del oyente en diferentes momentos del tiempo. Auditivamente se caracteriza por una prolongación a modo de "cola sonora". El tiempo de reverberación es una propiedad de las salas y se define como el lapso que debe transcurrir para que el sonido inicial se atenúe en 60 dB. Gran parte de la música grabada se escucha en pequeñas salas particulares con tiempos de reverberación muy cortos. Un dato curioso es que la mayoría de las grabaciones comerciales incorporan la reverberación y otros efectos que emulan la acústica de grandes salas y las hacen más gratas al oído. Las reverberaciones digitales han desbancado a los antiguos sistemas analógicos. La duración y la coloración tímbrica de esta cola dependen de la *distancia* entre el oyente y la fuente sonora y de la *naturaleza* de las superficies que reflejan el sonido.

  Los siguientes parámetros son los que condicionan el resultado de la reverberación. Manipulándolos podemos crear la sensación de tamaño de recinto y de posicionamiento de fuente y oyente dentro de él.

  - Tiempo de decaimiento: tiempo que tarda el sonido reverberado en disminuir 60 dB.
  - Retardo de las primeras reflexiones: en salas grandes las primeras reflexiones tardan en llegar más tiempo que en salas pequeñas.

  - Intensidad de las primeras reflexiones: determinada por la distancio del oyente y de la fuente sonora respecto a las superficies reflectantes.
  - Tipo de reverberación: tipo hall, tipo plate, tipo room... cada una de ellas proporciona una coloración diferente.
  - Densidad de las reflexiones: aumenta en función de la cantidad de trayectorias reflejadas que lleguen al oyente debido a que hay muchas superficies reflectantes.
  - Absorción selectiva de determinadas frecuencias: directamente relacionada con los materiales de las superficies reflectantes y puede simularse aplicando una determinada ecualización.

- **Eco:** es el efecto sonoro que se produce cuando un sonido rebota contra una superficie lejana y llega por duplicado al receptor con un cierto retardo. Antiguamente este efecto se conseguía gracias a los cabezales de grabación y reproducción de un magnetófono. Inyectando un sonido, grabándolo y reproduciéndolo inmediatamente obtendremos un retardo cuyo tiempo estará determinado por la distancia entre los cabezales y por la velocidad de la cinta. Actualmente esto se consigue mediante retardos digitales (delays) que permiten tiempos desde una milésima hasta tres o cuatro segundos.

  Además del tiempo de retardo se pueden manipular otros parámetros:

  - Regeneración: la señal retardada vuelve a retardarse.
  - Múltiples líneas de retardo: es posible retardar de maneras diferentes pero simultáneas una misma señal.
  - Panoramización: permite hacer sonar las repeticiones alternativamente en uno u otro lado del espacio acústico o ir desplazándolas progresivamente en una determinada dirección.

  Los retardos no sólo se utilizan para simular eco:

  - Retardo muy corto (menor de 30 milisegundos) y una cierta realimentación: alteración de la tímbrica. El sonido se hará metálico y adquirirá resonancias muy definidas en determinadas frecuencias.
  - Retardo entre 20 y 80 milésimas: afecta a la presencia del instrumento ya que sumamos perceptualmente dos sonidos iguales. Genera sensación de sonido mas grueso.

  - Retardos mayores de 80 o 100 milisegundos: el efecto principal que obtenemos es de tipo rítmico, por tanto hay que ajustar el tiempo de retardo al tiempo de la música.

Basados en filtrado

Cualquier algoritmo o proceso computacional que a partir de una entrada genere una salida, puede considerarse como un filtro digital.

Una función muy común de los filtrados son los ecualizadores. Un ecualizador consta de varios potenciómetros, cada uno de ellos asociado a una banda de frecuencia, que permiten amplificar o atenuar estos componentes frecuenciales. Permite como máximo manipular tres parámetros:

- Frecuencia de actuación o central: determinar sobre qué zona del espectro se quiere actuar.
- Anchura de banda o factor Q: determinar la región en torno a la frecuencia central sobre la que se actuará.
- Nivel de atenuación/amplificación: determinar la magnitud en dB que necesitamos realzar o atenuar en la banda sobre la que se actúa.

Basados en la amplitud

Se basan en la manipulación de la amplitud de la señal de audio. Los clasificamos en *simples* y *sofisticados*:

- Efectos simples
  - Modificar ganancia: multiplicación de cada muestra por un valor real. Si el valor está entre 0 y 1, el nivel sonoro disminuye. Si el valor es mayor que 1 el nivel sonoro aumenta.
  - Silenciar: multiplica por 0 la zona seleccionada.
  - Normalizar: es un caso particular de modificación de ganancia. Consiste en obtener la máxima amplitud posible sin que se produzca distorsión.
- Efectos sofisticados
  - Compresores: se utilizan para deducir el rango dinámico de una señal. Se utiliza mucho en la grabación de partes vocales, acentúa los mínimos muy débiles que presenta la voz. Se utiliza en guitarra eléctrica(típico efecto heavy).

- Limitadores: limitan la amplitud máxima. Suelen utilizarse en grabaciones de conciertos para evitar la saturación.
- Expansores: lo opuesto a un compresor. Acentúa los cambios. Se utiliza para realzar grabaciones antiguas que presentan un rango dinámico estrecho.
- Distorsión: define las pérdidas o degradaciones inevitables en una señal.

3.4. Alternativas de desarrollo encontradas

Este apartado se centrará en las diferentes opciones que se han encontrado para desarrollar el compositor musical teniendo en cuenta la información extraída de las secciones anteriores. La elección de las opciones pasa por diferentes fases. En un primer momento se realizó un estudio del lenguaje de programación que más se adecuaría al objetivo perseguido. Las principales opciones contempladas fueron C y Java, ambas por su extensión, potencia y flexibilidad. Tras realizar un análisis más profundo de las posibilidades que ofrecían ambos lenguajes, se decidió emplear Java para la implementación del sistema. Los motivos principales que llevaron a esta decisión fueron en primer lugar el mayor dominio de Java por parte del autor y que al ser un lenguaje más moderno, ofrece un mayor número de opciones en relación a aspectos multimedia en general.

Una vez decidida la plataforma sobre la que asentar el proyecto, se pasó a comprobar cuáles eran las opciones reales que ofrece Java para el manejo de sonidos. Para ello se contemplaron diferentes posibilidades mediante la búsqueda y pruebas en internet y los resultados más interesantes se muestran a continuación:

3.4.1. Librería javax.sound.midi

Uno de los primeros casos que fueron estudiados es el de la librería javax.sound.midi contenida en el API de Java. Esta librería se caracteriza por la presencia de interfaces y clases que permite el manejo de archivos .midi. Esto presenta algunas ventajas y bastantes inconvenientes.

La principal ventaja es que contiene únicamente las clases que son necesarias para manejar este tipo de archivos de sonido, lo que restringe y centra el ámbito del problema simplificándolo en gran medida. Existen clases como *Instrument*, *MidiEvent*, que se encarga de informar de los eventos contenidos en un archivo MIDI, u otras clases como *Sequence*, que actúa como una

estructura de datos que contiene información sobre un archivo musical y que suele contener uno o más objetos de la clase *Track*, cuyo objetivo es representar un flujo independiente de eventos MIDI que se pueden almacenar con otros tracks en un mismo archivo MIDI. Esta última característica permite de alguna forma atravesar la limitación de los MIDI de poseer únicamente 16 canales.

Existen varios inconvenientes que se pueden extraer de este sistema de trabajo. Uno de los principales es el hecho de tener que utilizar obligatoriamente archivos MIDI, ya que se limita bastante el número y la calidad de los sonidos generados. Además esto implica que, debido al enfoque de trabajo, venga acompañado de la utilización de bancos de sonido predeterminados. De esta manera se simplifica en gran medida el manejo de sonidos ya que utilizando las interfaces *Soundbank* y *Synthesizer* se permite reproducir los sonidos sin mucha dificultad, pero la música generada es muy dependiente del banco de sonido introducido.

Estos inconvenientes se presentan insalvables y por ello se decidió eliminar también la idea de utilizar este tipo de implementación, debido a que limita el tipo de sonido alcanzado, reduciendo las posibilidades de alcanzar música de una cierta calidad y armonía.

3.4.2. Librería sun.audio

Esta es una librería muy simple y fácil de manejar. Se encuentra en fase experimental y es muy poco conocida, utilizada y extendida. Su facilidad de comprensión y manejo viene compensada con grandes limitaciones en cuanto a las operaciones que permite realizar.

Trabaja con objetos pertenecientes a las clases *AudioPlayer* y *AudioStream*. Mediante el *AudioStream* se carga el fichero de sonido necesario y con *AudioPlayer* y sus métodos start() y stop() (que son los únicos que posee) permiten la reproducción del sonido.

Únicamente con estos elementos se ha podido construir un programa que, apoyado en *Threads*, permite cargar varios sonidos en arrays y reproducirlos simultáneamente controlando su tempo y todo ello en menos de cien líneas de código.

A continuación se presenta ese ejemplo para mostrar la sencillez de manejo que ofrece este sistema:

```
import sun.audio.*; import java.io.*; import
javax.sound.sampled.*;

/**
 * Clase encargada de la reproducción de las pistas que se corresponden con
 * cada uno de los instrumentos disponibles.
```

```
 *
 * @author Enrique Jiménez Domingo
 * @version 1.0
 *
 */

public class ReproducirPista extends Thread {

    int notas [];
    String ins=null;
    int duracion=0;

    /**
     * Constructor que inicializa los diferentes parametros.
     *
     * @param str   nombre del archivo .wav empleado
     * @param longitud  tamaño del array de notas
     * @param segundos  numero de segundos que debe durar la ejecución del instrumento correspondiente
     *
     */
    public ReproducirPista(String str, int longitud, int segundos)
    {
        ins=str;
        notas = new int[longitud];
        duracion= segundos;
    }


    /**
     * Método que rellena de unos y ceros el array de longitud predeterminada que se corresponden con las notas
     * que deben sonar o silenciarse.
     *
     * @return notas array de enteros con las notas a tocar o silenciarse.
     *
     */
    public int[] rellenarNotas()
    {
        for(int i=0;i<notas.length;i++)
        {
            notas[i] = (int)(Math.random()*2);
        }

        return notas;
    }

    /**
     * Método heredado de la clase Thread que reproduce cada array de notas.
     *
     * @exception e captura cualquier tipo de excepción
     *
     *
     */
    public void run()
    {
        String ruta= "C:\\j2sdk1.4.2_13\\bin\\Drumset2\\" + ins;
```

```
        for(int i=0;i<notas.length;i++)
        {
            try
            {
                int arrayNotas[] = rellenarNotas();

                InputStream in = new FileInputStream(ruta);

                if(arrayNotas[i]==1)
                {
                    AudioStream as = new AudioStream(in);
                    AudioPlayer.player.start(as);
                    Thread.sleep(500);
                }
                else
                {
                    Thread.sleep(1000);
                }
            }

            catch(Exception e){}
        }
    }

    public static void main(String args[])
    {
        new ReproducirPista("XLX_016.WAV", 120, 120).start();
        new ReproducirPista("XLX_022.WAV", 120, 120).start();
        new ReproducirPista("XLX_005.WAV", 120, 120).start();
        new ReproducirPista("XLX_031.WAV", 120, 120).start();
        new ReproducirPista("XLX_048.WAV", 120, 120).start();
        new ReproducirPista("XLX_007.WAV", 120, 120).start();
    }

}
```

Es posible comprobar como cada sonido diferente se carga en un *Thread* que reproduce su sonido, según las notas que se hayan creado en su array de notas, permitiendo de esta forma hacer sonar varios sonidos simultáneamente, controlando su tempo pasado como parámetro en cada uno de los *Threads*.

Su sencillez tiene como contraposición limitaciones importantes. Con este sistema únicamente se podría girar en torno a estos conceptos, debido a que no ofrece funcionalidades mas allá de las de cargar, reproducir y parar los sonidos. No se contemplan aspectos como los efectos (muy complicados de introducir con este sistema y que aumentaría enormemente las posibilidades creativas de las piezas) y el diseño de clases sería excesivamente simple, pudiendo ser demasiado limitado para los objetivos del proyecto. Aún así es una opción interesante y atractiva por la sencillez de manejo que permitiría centrarse más en los aspectos musicales dejando a un lado los problemas

tecnológicos.

3.4.3. Librería javax.sound.sampled

Este paquete incluido y detallado en el API de Java, ofrece una serie de interfaces y clases para poder desarrollar aplicaciones utilizando cualquier tipo de archivo de sonido, desde MIDI a WAV pasando por MP3.

El sistema que emplea esta librería es la utilización de líneas (Lines o DataLines) y de AudioStrems para cargar y mover los sonidos y realizar las acciones pertinentes. En principio este sistema puede parecer bastante sencillo, pero al enfrentarse al momento de trabajar con ello no lo es tanto. Estas clases precisan de mucha información adicional y del uso de objetos de otras clases para completar aquella información que poseen, lo que durante los primeros pasos dificulta mucho el avance de la implementación del sistema.

Otra interfaz importante que posee esta librería es la interfaz *Mixer*. Esta interfaz se encarga de gestionar las líneas de datos, pudiendo crear una o varias líneas y posteriormente reproducirlas, lo que deriva en un sistema con muchas entradas (cada una de las líneas de datos con archivos de sonido) y una única salida (la resultante de mezclar las distintas líneas en una sola). El problema que conlleva la implementación de esta interfaz es que obliga a utilizar otra de las interfaces, la interfaz Clip, cuyo manejo unido a todos los aspectos anteriores dificulta bastante la evolución del trabajo, ya que en muchos casos es necesario crear un gran número de objetos de cada clase para poder crear una pequeña parte de lo que será el programa. Esto puede repercutir en el coste y posiblemente más adelante, en la ralentización del sistema si se utiliza una gran cantidad de sonidos o líneas.

Estos inconvenientes presentados pueden dificultar en gran medida el desarrollo del compositor. Aún así esta opción es considerada atractiva por la buena gestión y división que realiza de las líneas de datos, permitiendo cargar un sonido en cada una de ellas para poder manejarlas más adelante de una forma libre e independiente del resto. Por ello sería interesante encontrar alternativas que permitan solventar los problemas expuestos.

3.4.4. Efectos

Como se ha comentado anteriormente, un añadido interesante al compositor musical que se pretende desarrollar es permitir la inclusión de efectos. Esto otorgaría muchísimas posibilidades al compositor automático y sería muy interesante observar como mediante las técnicas de inteligencia artifi-

cial se ajustan las notas de los instrumentos con cualquiera de los efectos añadidos a la melodía de una canción.

Para poder alcanzar esta meta, se han realizado varias búsquedas con el objetivo de encontrar un sistema que se pudiese acoplar al que se pretende desarrollar sin que afecte demasiado al diseño del mismo ni nos desvíe de los objetivos principales del proyecto. En este caso la búsqueda no ha sido tan fructífera como la anterior, en la que se buscaban librerías para implementar el sistema, ya que ahora únicamente se han encontrado dos formas de añadir efectos a un determinado sonido cargado por el programa:

1. Clase ReverbType

   Es una clase incluida en la librería javax.sound.sampled comentada anteriormente. El sistema que utiliza es el de variar una serie de parámetros numéricos que se relacionan con propiedades del sonido, como puede ser la intensidad, el retraso, etc... y pasárselo a una señal de audio para que los tome como valores correctos o por defecto a reproducir. Con ellos se pueden conseguir efectos tales como simular que el sonido procede de una cueva, de un garage, de un armario, etc. El gran problema de este sistema es averiguar el valor numérico de cada uno de los parámetros que proporcionan el efecto deseado, ya que en algunos casos se encuentra documentado, pero para dar matices propios al sonido habría que experimentar mediante prueba y error. Esto conllevaría mucho tiempo y además las posibilidades en este ámbito se verían muy limitadas, a pesar de su aparente sencillez de manejo.

2. A través de filtros

   Java ofrece diferentes clases que permiten la inclusión de filtros para modificar los datos que sean necesarios proporcionándoles propiedades adicionales. Este sistema se basa en la idea de una vez obtenido el sonido al que se quiere añadir el efecto, se pase a través de un determinado filtro que lo modifique de la forma deseada. La idea principal es sencilla, pero la implementación no lo es tanto. Ejemplos de este tipo de clases que nos ofrece Java son *FilterInputStream*, *FilterChain* o *FilterConfig*, que vienen acompañados de la interfaz Filter.

El tipo de efectos que se incluirá dependerá de cual de las opciones comentadas se adapte mejor al sistema. Este apartado se ha dejado abierto y no se planea implementar en el sistema actual. Si bien, a la hora de diseñarlo se ha tenido en cuenta la futura incorporación de efectos sin que ello deba afectar al resto de las funcionalidades del compositor.

3.4.5. Tipo de implementación elegida

Una vez estudiadas en profundidad las diferentes opciones planteadas y habiendo realizado pruebas con cada una de ellas, sopesando sus pros y contras, se ha decidido que la implementación del sistema se realice empleando la librería javax.sound.sampled pero salvando las dificultades que se comentaron en su momento.

Recordando, la principal dificultad era la uso de la interfaz *Mixer* que gestionaba cada una de las líneas de datos y que obligaba a utilizar *Clips* para cada uno de los sonidos. Esto derivaba en la necesidad de crear un gran número de objetos que producían un gran coste y añadían un alto grado de dificultad a la programación.

Haciendo un estudio paralelo para poder solucionar estos problemas, se encontró una solución sencilla y elegante que permitirá un correcto desarrollo del proyecto y mejorará las condiciones que planteaba inicialmente este sistema. Lo que se hará para gestionar las líneas de datos y cada uno de los sonidos será introducir cada una de ellas en *Threads*. De esta manera se podrá manejar de forma independiente cada hilo, pudiendo gestionar sus operaciones de manera adecuada en el momento que sea necesario y con la misma posibilidad de hacer funcionar varios en un mismo momento, con lo que se consigue la mezcla se sonidos.

Una mejora que se podría conseguir a esta implementación, es la utilización de, en lugar de los hilos clásicos, los llamados *ThreadPool*. Ésta es una técnica que permite agrupar los *Threads* en una única estructura, pudiendo manejar varios a un mismo tiempo, mejorando en gran medida las prestaciones, ya que su uso puede llegar a ser incluso cien veces más rápido que el de los *Threads*. El principal inconveniente del *ThreadPool* está en que no es ilimitado (por defecto tiene 25 slots por procesador). Si se agotan los *Threads* podríamos provocar que la aplicación quedase en un estado de Idle infinito (o deadlock). Además de este importante problema, el alcance del proyecto no precisa de la necesidad de manejar este tipo de estructuras ya que las entradas empleadas son, a priori, sencillas y de igual manera se presenta innecesario el hecho de manejar conjuntos de hilos de forma simultánea, ya que es más recomendable manipular cada hilo de manera independiente. Aún así, será siempre una opción a tener en cuenta para futuros desarrollos.

Capítulo 4

Diseño

A continuación se presenta el diseño de la solución del problema que garantiza la viabilidad y el rendimiento positivo de la solución. Para lograrlo se emplearán las técnicas necesarias que permitan alcanzar los objetivos planteados de la forma más adecuada y reutilizable posible.

4.1. Diseño de la solución

Como es bien sabido, el objetivo del sistema es el de permitir realizar composiciones musicales de una manera sencilla para el usuario. Hasta ahora se han dado respuesta a los principales problemas tecnológicos encontrados. En esta sección se detallarán las diferentes decisiones de diseño tomadas y la forma en la que han sido implementadas.

El diseño desarrollado deberá cumplir los siguientes requisitos:

- Deberá realizarse teniendo en cuenta los objetivos que se han de cumplir y también que estos deberán alcanzarse independientemente del tipo de implementación elegida, es decir, la implementación no debe limitar o reducir el ámbito del sistema.
- La implementación debe realizarse de tal forma que se minimice el impacto producido por cualquier modificación posterior en el análisis o diseño, ya que es muy probable que se introduzcan nuevas funciones en el futuro.
- Se deberá implementar una interfaz sencilla y usable que permita a un usuario con escasos conocimientos, tanto informáticos como musicales, manejar el sistema con facilidad.

- Deberá realizarse un amplio abanico de pruebas que aseguren el correcto funcionamiento de cada una de las funcionalidades del sistema y un control de errores detallado.
- Una vez que las pruebas hayan sido satisfactorias, se desarrollarán los diferentes aspectos del algoritmo genético y se aplicarán al problema que se presenta.

El primer paso para el desarrollo completo del sistema fue el de diseñar de manera esquemática el diagrama de clases al que se debía ajustar. Tras esto, comenzaron a añadirse las distintas funcionalidades existentes y a repartirse en las clases correspondientes hasta obtenerse el diseño de clases final que se presenta en la figura 4.1.

En este diseño se muestran sólo los métodos y funciones principales del sistema por motivos de extensión y legibilidad. Para una visión más detallada de todos los componentes del sistema, se puede consultar el javadoc completo en el material adjunto a este documento.

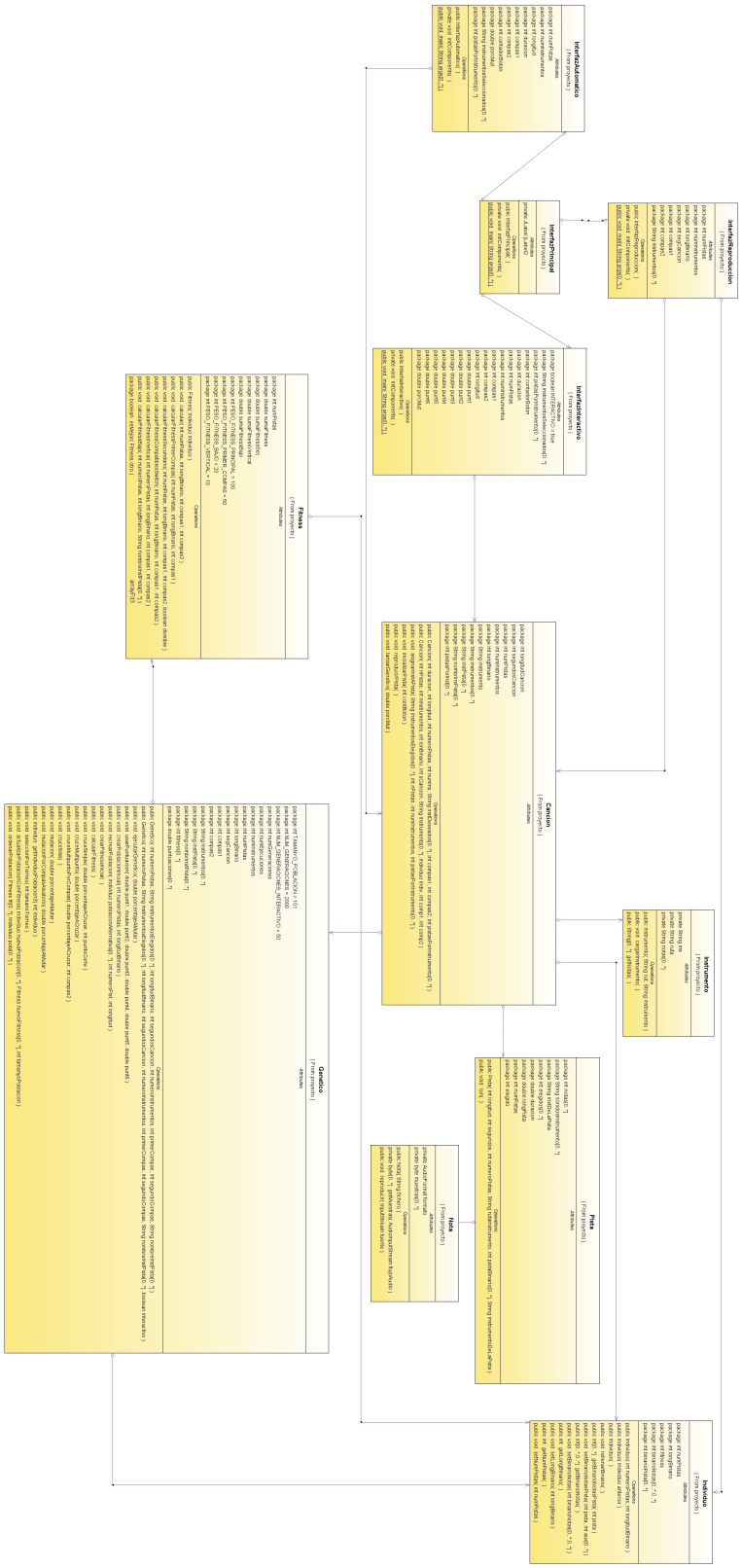

Figura 4.1: Diagrama de clases completo del sistema

Como es posible comprobar se ha realizado una clara división de las distintas partes que forman el compositor musical de manera que quede todo encapsulado y estructurado de la forma más adecuada posible pensando en la reutilización y posibles modificaciones posteriores.

En la siguiente sección se explican con más detalle las principales clases y métodos que forman el sistema. Si bien, toda la parte concerniente al algoritmo genético, su construcción, funcionamiento, versiones y mejoras se comentará ampliamente en el próximo capítulo.

4.2. Funcionalidades y clases

En esta sección se explican los aspectos más importantes del diseño del sistema y se detallan las clases y funciones más importantes de las que consta.

4.2.1. Nota

Es la clase que se encarga de reproducir el sonido correspondiente en cada nota de la pista que se esté reproduciendo.

Los métodos más relevantes son:

- getMuestras: permite obtener un array de bytes que contiene el flujo del audio que se debe reproducir.
- reproducir: permite introducir la información del audio en una línea de datos y ser reproducida por el sistema.

Los métodos de esta clase serán llamados desde la clase Pista que se detalla a continuación.

4.2.2. Pista

Esta clase encapsula las diferentes pistas que posee cada canción. Cada pista está compuesta por un array binario en el que el "1" representa la reproducción del sonido de una nota y "0" la ausencia de sonido. Cada pista lleva asociado un sonido concreto de un determinado instrumento. Además esta clase extiende de la clase *Thread* para de esta manera permitir que las diferentes pistas de una misma canción se puedan reproducir simultáneamente.

El único método que contiene esta clase es:

- run: método heredado de la clase *Thread* en el que se recorre el array binario de notas y se llama a la clase Nota cuando se encuentre un '1' en dicho array y por tanto se deba reproducir el sonido correspondiente.

También se controla aquí el tiempo de la canción, es decir, se adapta la velocidad de reproducción de las notas en función de las notas que se deban reproducir y de la duración de la canción.

4.2.3. Instrumento

Representa el instrumento que será tocado para cada pista. Dentro de cada instrumento existen numerosos sonidos. Al asignar un instrumento a una pista sólo se está eligiendo el instrumento no el sonido concreto, que será seleccionado al azar por el sistema.

El método principal de esta clase es:

- cargarInstrumento: este método se encarga de cargar todos los sonidos existentes de un mismo instrumento en un array para que puedan, más adelante, ser seleccionados, y ser asignados a una pista concreta. De esta manera podrán reproducirse cuando corresponda.

En esta clase es necesario pasar la ruta en la que se encuentran los instrumentos con el fin de que el sistema encuentre aquellos que van a ser seleccionados y asignados.

4.2.4. Individuo

Esta clase gestiona todas las operaciones relacionadas con la creación y manipulación de los individuos, proporcionando funciones que permiten obtener información acerca de los individuos que forman la población o modificar la existente.

Los métodos más utilizados e importantes de la clase Individuo son:

- rellenarBinario: este método se encarga de rellenar el array binario de notas de manera aleatoria. Sólo se utilizará para la primera población, ya que ese array se irá modificando posteriormente con la ejecución del algoritmo genético.
- getBinarioNotasPista: permite recuperar el valor del array binario de notas de una determinada pista. Éste es uno de los métodos más utilizados por el sistema ya que es necesario obtener su valor en varios puntos de la ejecución del sistema.

- setBinarioNotasPista: permite modificar el array de notas de una pista concreta de un individuo. Este método es llamado en cada ejecución del algoritmo genético para poder almacenar las modificaciones realizadas por el mismo sobre los individuos que componen la población.

Además de la funcionalidad comentada, resaltar la existencia de distintos tipos de constructores para esta clase que permiten inicializar un individuo desde cero o crearlo a partir de uno ya existente además de el constructor por defecto.

4.2.5. Fitness

Como su propio nombre indica, esta clase se encarga de encapsular las diferentes funciones de fitness que se utilizarán en el sistema y los métodos asociados a ellas.

Las funciones de fitness desarrolladas persiguen diferentes objetivos y proporcionan un valor numérico adaptado a lo cerca o lejos que se encuentren de él permitiendo que el algoritmo genético evolucione favorablemente. A lo largo de todo el proceso de creación del sistema se ha implementado y experimentado con diferentes objetivos y enfoques, buscando diversos resultados para cada uno de ellos.

En la siguiente sección se explicarán en detalle cada una de estas funciones y se proporcionará toda la información necesaria para el completo entendimiento del funcionamiento y de las conclusiones que se pretenden extraer de todas y cada una de ellas.

4.2.6. Genético

La clase Genético contiene toda la funcionalidad propia del algoritmo genético así como algunos métodos auxiliares necesarios para su ejecución. Se encuentran incluidos aquí todos los operadores genéticos implementados en sus diferentes versiones.

Los métodos más importantes del sistema empleados en esta clase son:

- ejecutarGenetico: es el método que se encarga de ir llamando sucesivamente al cálculo del fitness y cada uno de los operadores genéticos tantas veces como sea el valor del número de generaciones deseadas.

- crearPoblacionInicial: desde este método se crea la población con la que comenzará a ejecutarse el algoritmo genético. Se inicializa cada uno de

los individuos de la población y se llama al método correspondiente de la clase Individuo para rellenar el array binario de notas de manera aleatoria.

- calcularFitness: es la parte principal de todo algoritmo genético. Desde aquí se pasará a conocer cómo de bueno es cada uno de los individuos de la población con respecto al objetivo que se quiere alcanzar. Como se comentó anteriormente, los detalles de cálculo de las distintas funciones de fitness se explicarán en detalle en el capítulo siguiente.

- seleccionPorTorneo: operador de selección clásico en el que se selecciona un número determinado de individuos para que el que posea un mayor valor de fitness sea pasado a la siguiente población para pasar por el resto de operadores genéticos.

- cruceSimple: operador de cruce que intercambia la primera mitad de un individuo con la primera mitad de otro (ver figura 4.2).

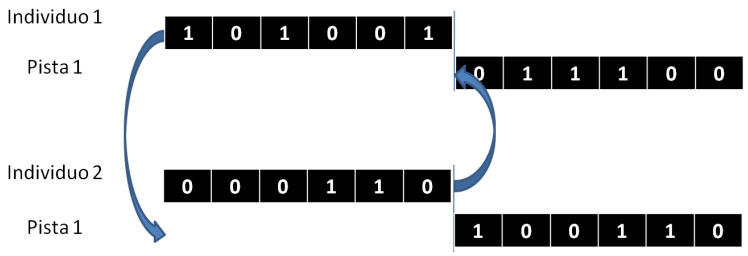

Figura 4.2: Cruce Simple aplicado a pistas de dos individuos

- cruceMultipunto: operador de cruce cuyo funcionamiento consiste en seleccionar dos puntos aleatoriamente, y la parte del individuo que quede entre esos dos puntos se intercambiará con la misma parte del otro individuo a cruzar. Los puntos de corte van cambiando en cada uno de los cruces realizados (véase figura 4.3).

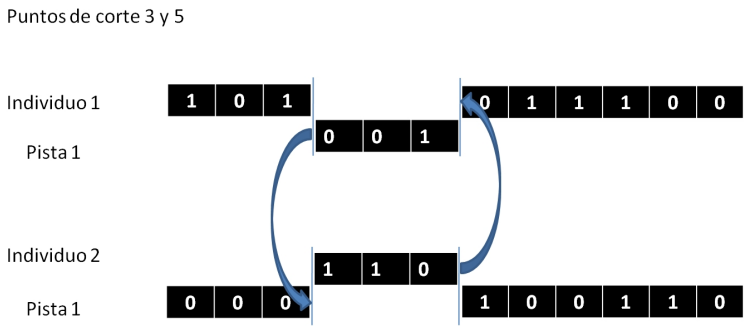

Figura 4.3: Cruce multipunto aplicado a pistas de dos individuos

- cruceMultipuntoPorCompas: operador de cruce que selecciona un compás de la composición de manera aleatoria y lo intercambia con otro compás de otro individuo seleccionado también de forma aleatoria (figura 4.4).

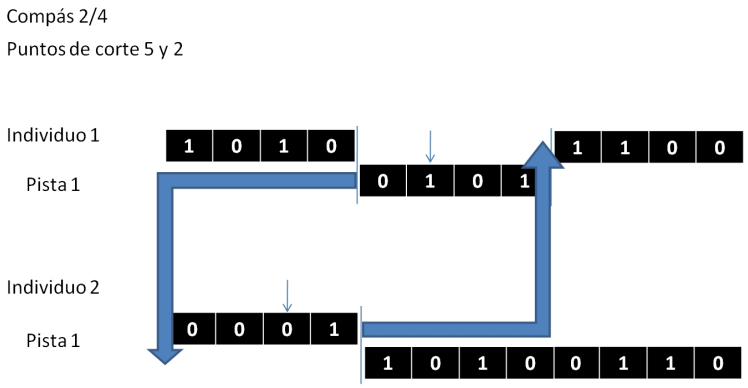

Figura 4.4: Cruce multipunto por compás aplicado a pistas de dos individuos

- mutación: se encarga de mutar el array binario de notas cuando el valor de un número aleatorio sea menor que un valor umbral establecido.
- mutacionPorCompasAleatorio: operador de mutación que se encarga de seleccionar un compás al azar y cambiarlo por otro que se genera en ese instante de manera aleatoria (figura 4.5).

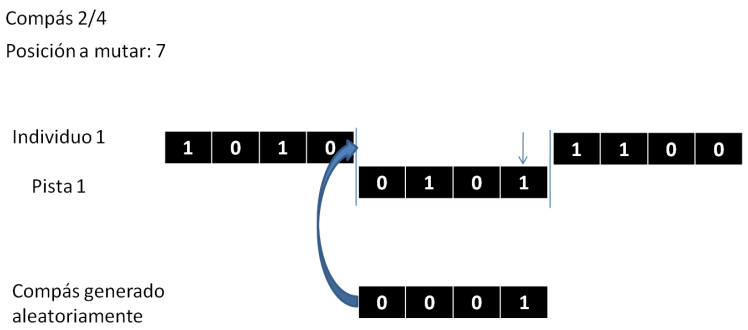

Figura 4.5: Mutación por compás aleatorio aplicado a pista de un individuo

Como se puede comprobar, se han implementado varias versiones de los operadores genéticos convencionales con el fin de estudiar las diferencias que existen entre ellos y analizar los resultados de las pruebas con las diferentes combinaciones que pueden realizarse para de esta forma establecer unas conclusiones adecuadas y válidas, que puedan hacerse extensibles a otros tipos de sistemas similares al planteado en este proyecto.

4.2.7. Canción

Esta es la clase central del sistema. Desde aquí se crean los diferentes componentes del sistema, desde la creación de las pistas, la asignación de instrumentos y la reproducción de los sonidos, hasta el lanzamiento del algoritmo genético sobre una población de individuos creada. Por ello la clase Canción encierra lo que su propio nombre indica, toda la información necesaria para que el sistema sea capaz de componer una canción.

A continuación se resume la funcionalidad que poseen los métodos más significativos de esta clase:

- asignarInstAPista: este método es el encargado de, en base a los deseos plasmados por el usuario, seleccionar un sonido concreto de un instrumentos y asignarlo a una de las pistas de la canción. La asignación de ese sonido a la pista se mantendrá durante toda la composición.

- inicializarPista: se encarga de crear una pista asignándole el array binario de notas con el instrumento correspondiente, para que posteriormente pueda ser reproducida.

- reproducirPista: es la función que se encarga de hacer sonar de manera simultánea todas las pistas de una canción. Consiste en una llamada al

método *start* heredado de la clase *Thread* (ya detallada en el capítulo) anterior, que ejecuta el método *run* de la clase Pista con la información precisa en cada caso.

Además de los métodos comentados, también se incluyen en esta clase los métodos que guardan en fichero las canciones creadas y las poblaciones que el usuario desee guardar para continuar trabajando con ellas posteriormente, así como los métodos correspondientes para cargar dicha información.

Cabe recordar, que si se desea obtener una información más completa de las clases y métodos que componen todo el sistema, se puede conseguir estudiando la documentación adjunta a este proyecto con el javadoc completo de la aplicación.

4.3. Interfaz de usuario

Para facilitar la interacción del usuario con el sistema, se ha realizado un interfaz de usuario desde el cual se podrán configurar y gestionar las diversas partes del compositor en sus diferentes versiones.

La principal característica de la interfaz es la sencillez. El sistema está enfocado a que pueda ser utilizado por usuarios sin conocimientos musicales avanzados, por lo que la sencillez y facilidad de uso son objetivos primordiales.

A continuación se irán mostrando las capturas de las diversas partes del sistema y se irán comentando las funcionalidades existentes en cada una de ellas.

Ventana principal

La figura 4.6 muestra la pantalla inicial que encuentra el usuario al arrancar el sistema.

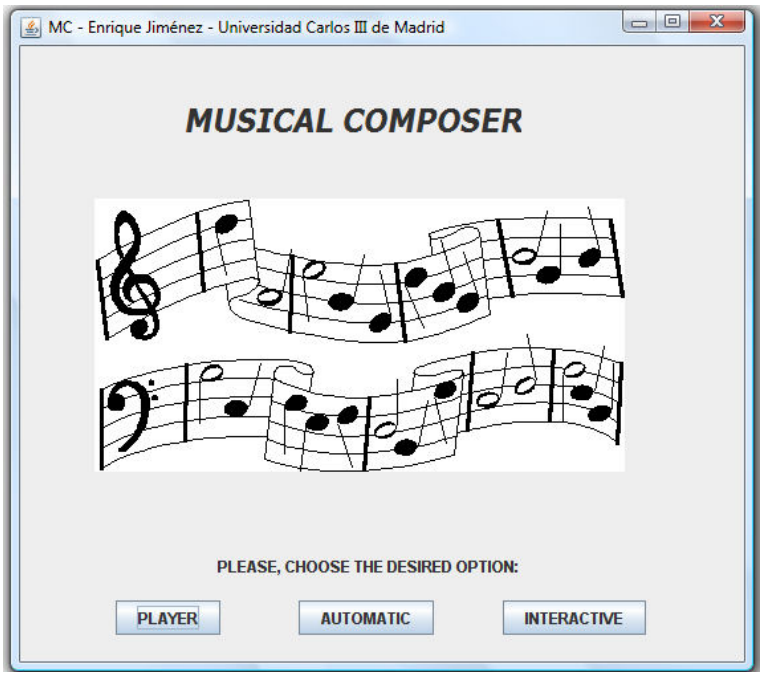

Figura 4.6: Interfaz principal del sistema

Como se puede observar, es una pantalla de inicio muy sencilla en la que el usuario podrá elegir la versión del sistema que desea ejecutar:

- **Automatic**: se accede al modo de creación automático del sistema.
- **Interactive**: da paso a la pantalla de creación interactiva del sistema.
- **Player**: permite el acceso al reproductor de canciones previamente generadas y guardadas.

Sistema de creación automático

Si el usuario desea ejecutar la versión automática del sistema se encontrará con la siguiente pantalla:

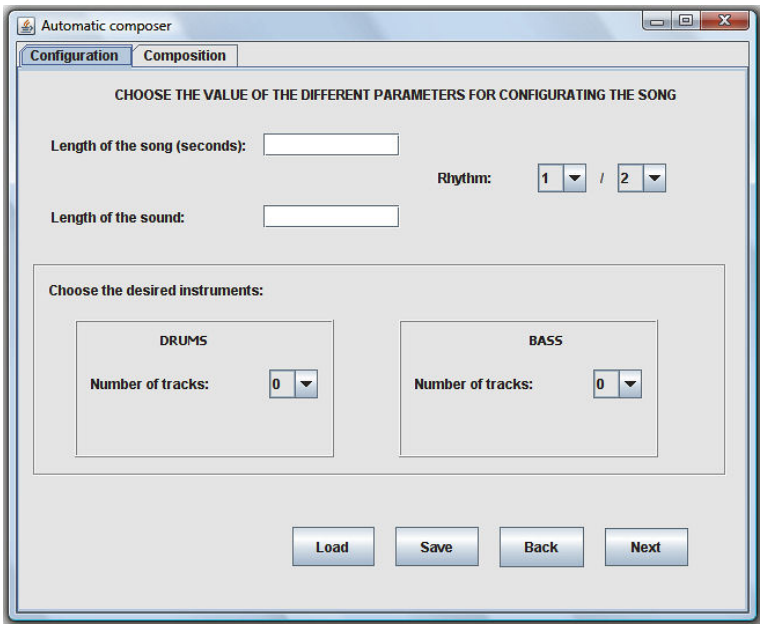

Figura 4.7: Interfaz versión automática. Pestaña de configuración

La figura 4.7 ilustra el primer paso a completar para realizar el proceso de composición. La pantalla se compone de dos pestañas. La primera de ellas es la de *configuración*(configuration). En ella se establecerá el valor de los diferentes parámetros que posteriormente poseerá la canción generada. Como se puede comprobar, la información que se le pide al usuario es muy clara y para introducirla no son necesarios conocimientos musicales o informáticos avanzados.

Esta pantalla posee también cuatro botones diferentes en su parte inferior. Las funcionalidades de estos botones son las siguientes:

- **Load**: este botón permite cargar un fichero de texto que posea una configuración para el sistema que haya sido previamente almacenada. De esta manera el usuario no tiene que introducir los datos, aparecerán automáticamente al cargar el fichero.

- **Save**: como su propio nombre indica, guarda una configuración para que pueda ser utilizada posteriormente. Los parámetros que estén en pantalla cuando se accione el botón, serán almacenados en un fichero de texto.

- **Back**: la funcionalidad de este botón es regresar a la pantalla inicial.

De esta manera se puede volver a elegir directamente la versión del sistema a utilizar.

- **Next**: se encarga de pasar a la siguiente fase del proceso de composición. Para poder acceder a la pestaña de *composición* se deben rellenar los campos necesarios. En caso de no cumplir este requisito el sistema mostrará el error correspondiente. La figura 4.8 muestra un ejemplo del tipo de errores que se mostrarán.

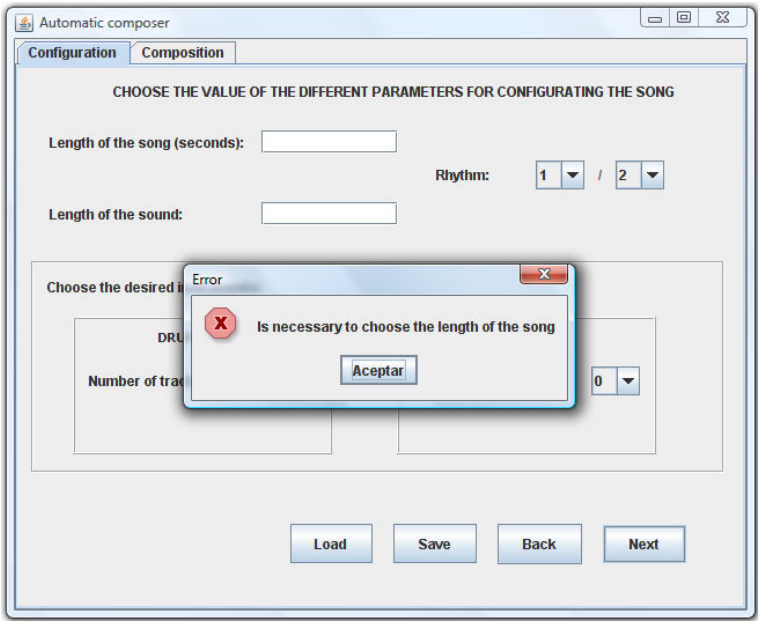

Figura 4.8: Interfaz versión automática. Ejemplo de control de errores

Una vez introducidos todos los datos previos correctamente, se accede a la pestaña de *composición*. Es en esta pestaña donde se escucharán las canciones iniciales y donde se dará paso a la ejecución del algoritmo genético que irá mejorando las composiciones. Todo ello se muestra en la figura 4.9:

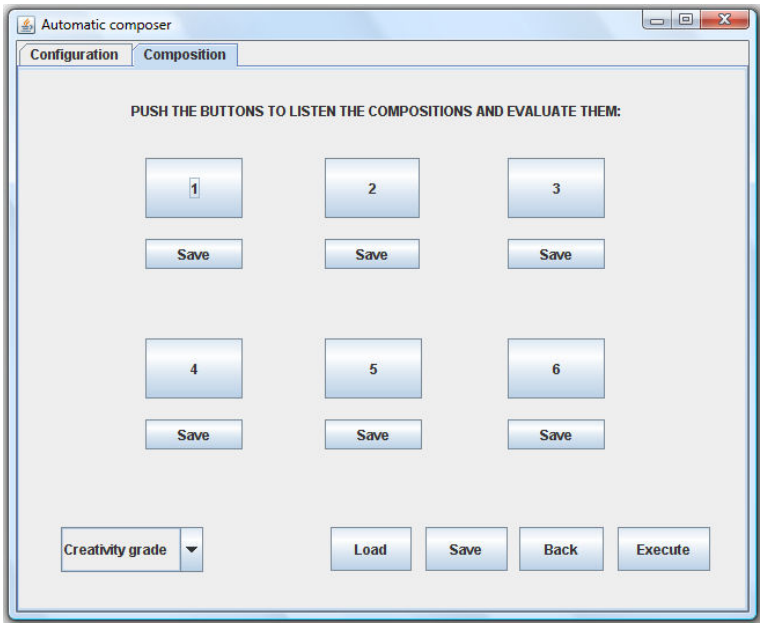

Figura 4.9: Interfaz versión automática. Pestaña de composición

Nuevamente aparece una ventana sencilla en la que se muestran con claridad los pasos a seguir para la composición. Existen seis grandes botones situados en el centro de la pantalla, cada uno con su número correspondiente. Cada uno de esos botones reproducirá una canción diferente de las que se encuentran en la población. Las que se escuchen antes de la ejecución del algoritmo genético serán composiciones aleatorias correspondientes a la población inicial, pero de esa forma se podrán observar los cambios sufridos tras la evolución. Para que se cree la población es necesario reproducir al menos una de las canciones reflejadas en los seis botones, en caso contrario se mostrará un error.

Una vez que se haya realizado esa acción, el usuario puede elegir entre las siguientes opciones que se corresponden con los botones situados en la parte inferior de la ventana:

- **Load**: permite cargar una población previamente guardada. De esa forma si el usuario abandonó una población sin que acabase de trabajar con ella y quiere retomarla, puede hacerlo empleando esta opción.
- **Save**: guarda la población actual para que pueda ser utilizada posteriormente. De esta manera el usuario puede dejar de trabajar en cualquier momento y retomarlo en el punto en el que lo dejó.

- **Back**: al pulsar este botón el usuario regresa a la pestaña de *configuración*. Es importante aclarar que al pulsar este botón se eliminará la población con la que se estaba trabajando, así que es recomendable que en caso de duda, se salve la población con la que se está trabajando antes de cambiar la configuración del sistema.

- **Execute**: mediante este botón se lanza la ejecución del algoritmo genético. Gracias a él se evoluciona la población de individuos para obtener composiciones adecuadas a los parámetros deseados.

Un aspecto no mencionado hasta ahora es la existencia del *grado de creatividad*. El valor seleccionado aquí influirá en el algoritmo genético, de forma que un menor grado de creatividad hará que la composición resultante se ciña más estrictamente a las reglas fijas marcadas, y un mayor grado provocará que la composición tenga más libertad, admitiendo mayores variaciones en las reglas. Remarcar que a mayor grado de creatividad mayor es la probabilidad de que las composiciones generadas tengan menos orden y por tanto al reproducirse no produzcan el efecto deseado. Si el usuario decide no seleccionar ningún valor, se tomará como cero.

Además de las funcionalidades mencionadas, existe también la posibilidad de guardar las canciones correspondientes a cada uno de los botones numerados. El botón *save* situado debajo de cada uno de ellos permite realizar esta acción. De esta manera, las canciones almacenadas podrán ser reproducidas posteriormente tantas veces como se desee utilizando la versión *reproductor* del sistema.

Sistema de creación interactivo

Si el usuario desea acceder a la versión interactiva del sistema se encontrará con la ventana que muestra la figura 4.10:

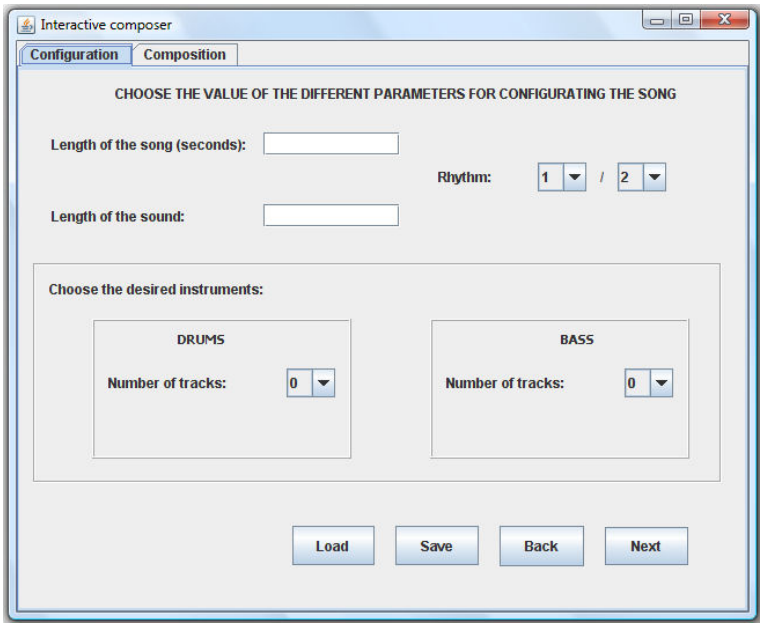

Figura 4.10: Interfaz versión interactiva. Pestaña de configuración

Como se puede comprobar, esta ventana es igual que la existente para el caso automático. Esto se debe a que los datos precisados en las dos versiones son los mismos y de esta manera se mantiene el mismo diseño para ambos casos. Por ello no se proporcionará información adicional sobre la pantalla. Para conocer la funcionalidad concreta de cada uno de los botones que aparecen consultar la sección 4.3.

Sin embargo, aunque la pestaña de *configuración* sea la misma para las dos versiones, no ocurre así para la parte de *composición*. La figura 4.11 muestra su aspecto:

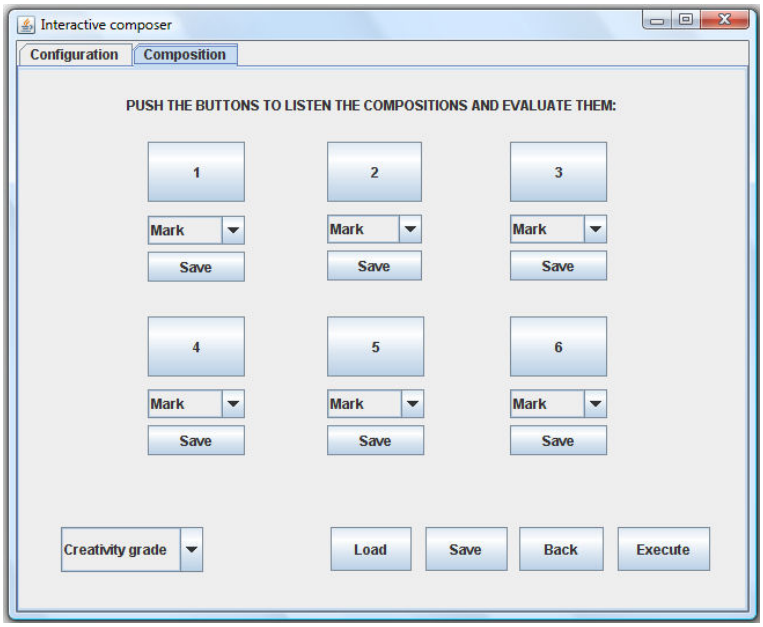

Figura 4.11: Interfaz versión interactiva. Pestaña de composición

Es fácil observar las grandes similitudes con el caso anterior, pero se aprecia un detalle que influye de forma determinante en la evolución de los individuos de la población. Debajo de cada uno de los botones numerados aparece un menú adicional. Este menú será el que permita al usuario otorgar la puntuación deseada a cada uno de las canciones aplicadas a los botones.

Las puntuaciones que podrá otorgar el usuario a cada una de las composiciones será la siguiente:

- Excelente (*Excellent*)
- Buena (*Good*)
- Normal (*Normal*)
- Mala (*Bad*)
- Muy mala (*Very bad*)

En función de la puntuación se dará una mayor o menor peso a las composiciones, ajustándose de esta manera a los gustos del usuario. Si el usuario no otorga puntuación a alguna de las composiciones, ésta será considerada

como mala y por tanto sus características no serán tan importantes a la hora de evolucionar la población.

Una vez que el usuario haya escuchado las composiciones que desee y las haya puntuado, se ejecutará el algoritmo genético teniendo en cuenta esas puntuaciones para dirigir en cierto modo el sentido de la evolución. Con esta característica añadida se establece la principal diferencia entre el sistema de creación automático y el interactivo.

El resto de botones continúan teniendo las mismas funcionalidades que se comentaron en la sección 4.3.

Sistema reproductor

Por último se presenta el reproductor del sistema. Esta parte se hacía necesaria para poder escuchar las composiciones que se habían generado mediante el sistema. El objetivo de esta pantalla es el de simplificar esa acción, proporcionando una manera cómoda de reproducir las canciones que el usuario ha deseado guardar.

La figura 4.12 ofrece una visión de este reproductor:

Figura 4.12: Interfaz sistema reproductor musical

La funcionalidad del sistema queda encerrada en los tres botones existentes en la parte inferior de la ventana:

- **Back**: este botón permite al usuario regresar a la pantalla principal del sistema y elegir la versión del mismo se desea ejecutar.
- **Load Song**: al pulsar este botón se abre un menú que permitirá cargar al reproductor un fichero de texto que contiene la información de una canción previamente creada y almacenada por el sistema.
- **Play Song**: como su propio nombre indica, este botón permite reproducir la canción previamente cargada.

Una vez que el usuario seleccione la canción que desee cargar, aparecerán en los campos situados en el centro de la ventana, los datos correspondientes a la misma. De esta forma se podrá comprobar si la canción cargada es la deseada y también si los parámetros marcados son correctos.

Capítulo 5

Soluciones propuestas aplicando técnicas de IA a la composición musical

En esta sección se detallarán todos los aspectos relacionados con el algoritmo genético, los diferentes enfoques y, por tanto, las diferentes funciones de fitness desarrolladas, así como otros elementos utilizados que permiten el funcionamiento correcto del sistema.

El objetivo del algoritmo genético a implementar es el de proporcionar una serie de individuos que, a partir de una población inicial, superen a individuos de generaciones anteriores hasta llegar a aquellos que representen composiciones musicales adaptadas a los deseos del usuario.

Los principales requisitos que debe cumplir el algoritmo genético son:

1. Requisitos de implementación:
    - La implementación del algoritmo genético debe generar una serie de resultados que en todo momento puedan ser consultados, es decir, debe generar información no volátil. De esta manera será posible ejecutar pruebas con diferentes parámetros y estudiar posteriormente los resultados obtenidos.
    - Deberá programarse el algoritmo de la forma más eficiente y eficaz posible. Es decir, el algoritmo tiene que funcionar correctamente, resolviendo todas las cuestiones para las que ha sido implementado, pero siempre buscando la máxima eficiencia y consumiendo el mínimo de recursos.

- Cada ejecución del algoritmo genético debe ser independiente de las anteriores. Para que las pruebas puedan ser interpretadas de manera adecuada no deben mezclarse las generaciones de unas ejecuciones con otras.
- Se podrá consultar la información referente a las poblaciones que se van generando en tiempo real. Esto es debido a que algunas ejecuciones pueden tardar un tiempo muy elevado y debe ser posible analizar los resultados a medida que se producen.

2. Requisitos de organización

   - Dentro de la población los individuos se ordenan por fitness. Así se conocerá siempre cuáles son los mejores individuos de una determinada población sólo con consultar su posición en la población.
   - Deberán poder guardarse individuos concretos en los ficheros para posteriormente tener acceso a ellos y que puedan ser reproducidos por el sistema.
   - Será necesario poder guardar una población en cualquier momento para que pueda ser cargada y evolucionada mediante el algoritmo genético en el momento que se desee.

3. Requisitos de funcionalidad:

   - La estructura de un individuo debe ser tan sencilla como el sistema permita, de modo que sea fácil manipular la información que contiene. Esto es debido a que en futuras implementaciones puede ser necesario añadir nuevas funcionalidades o mejorar las ya existentes.
   - Cada gen del individuo indica si en ese instante se debe reproducir una nota o no. Haciéndolo extensible a cada una de las pistas que éste posea, se produce la reproducción simultánea de cada una de ellas originando una canción.
   - Dado que el presente dominio admite numerosas variaciones y enfoques, la evaluación de los individuos se realizará desde distintas perspectivas tratando de obtener resultados adecuados y diferentes en cada caso. En este punto, se hará especial hincapié en el aspecto experimental, para estudiar los resultados de pruebas que puedan resultar de especial relevancia.

4. Requisitos de evolución:

- El sistema irá evolucionando según los siguientes pasos:
  - En cada ejecución se inicializará a 0 el valor de fitness de los individuos.
  - Se calculará el fitness de cada individuo según la función apropiada.
  - Se ordenará la población de individuos para establecer cuáles son mejores y peores.
  - Se realizará la selección por torneo de 5 individuos. Nótese que los individuos para el torneo son seleccionados aleatoriamente de modo que algunos de los individuos pueden ser seleccionados en más de una ocasión mientras que otros pueden no seleccionarse y, por tanto, perderse. Para evitar la pérdida del mejor individuo se ha utilizado *elitismo* de forma que el mejor de los individuos pasará siempre de una generación a la siguiente sin ser modificado.
  - Se ejecutará el tipo cruce elegido (simple, multipunto o por compás) de los individuos consecutivos de la población. Así se asegurará que todos los individuos son cruzados.
  - Se aplica la mutación deseada a la población para asegurar la variabilidad genética y evitar el estancamiento. El porcentaje de población que se mutará será introducido por el usuario a través de la interfaz.
  - Por último se calcula el fitness y se vuelve a ordenar la población.
- Este proceso se repite tantas veces como generaciones se deseen ejecutar obteniendo como resultado individuos cada vez más válidos y adecuados a las preferencias del usuario.

Una vez establecidos los requisitos y funcionamiento básicos del algoritmo genético sobre el que se asienta el sistema, se detallan a continuación los diferentes enfoques planteados y, por consiguiente, las funciones de fitness asociadas a cada uno de ellos.

El sistema se presenta en dos versiones claramente diferenciadas. La primera se trata de una vertiente cuyo objetivo es la composición automática por parte del sistema en base a una información inicial. La segunda realiza una composición interactiva, es decir, una composición en la que el usuario puede escuchar una serie de canciones generadas e ir asignando una puntuación a cada una de ellas. En base a esa puntuación el sistema dará más peso a las composiciones que más se asemejen a las elegidas por el usuario.

En primera instancia los sonidos que se emplearán para la búsqueda de composiciones rítmicas será únicamente aquellos pertenecientes a la batería. Esto es debido a que este instrumento permite desarrollar ritmos que sirvan de base en cualquier composición y además resulta más fácil su reconocimiento y utilización por parte del usuario.

Seguidamente se comentan ambas versiones y las diversas funciones de fitness aplicadas a cada una de ellas.

5.1. Sistema de creación automático

Como se mencionaba anteriormente, esta versión del sistema se centra en la composición de música de forma automática, es decir, sin interacción del usuario a excepción de la elección de los parámetros iniciales. Esto se puede llevar a cabo gracias al desarrollo de funciones que hacen que la evolución de las composiciones se produzca en base a unos cánones predefinidos, llegando a obtenerse canciones con orden y sentido.

Es bien sabido que la función de fitness es la parte fundamental de cualquier algoritmo genético ya que en función de sus valores se evoluciona en un sentido u otro. El desarrollo de cualquier función de fitness debe tener en cuenta aquellos factores que puedan y deban influir en los resultados deseados y adaptarla de forma que el sistema sea capaz de reconocer esa información y controlar la secuencia hacia esa dirección. La creación de esta función suele ser la parte más complicada de este tipo de sistemas debido a que, como se comentaba anteriormente, las variables a tener en cuenta pueden crecer de manera considerable. Si trasladamos este problema al mundo del arte y de forma más concreta a la música, es sencillo observar la gran variedad de estilos, ritmos, configuraciones y posibilidades que aparecen. Por ello, la función de fitness a elegir e implementar es una tarea especialmente complicada en este ámbito.

Ante este gran abanico de posibilidades, se decidió comenzar con una función de fitness sencilla e ir añadiendo cada vez mayor complejidad, controlando más variables y aspectos musicales, que llevasen a las composiciones obtenidas de forma progresiva a un nivel más alto.

En los siguientes apartados se detallarán estas funciones y las diferentes modificaciones y mejoras que se han ido introduciendo para tratar de alcanzar composiciones más ordenadas y correctas.

5.1.1. Fitness basado en patrones

La primera función de fitness que se ha desarrollado se encarga de analizar cada una de las pistas que componen la canción buscando patrones que puedan repetirse en la canción, para que de esta manera se reproduzca una misma secuencia de sonidos de forma constante otorgando orden y ritmo a la composición.

Para implementar este tipo de fitness el procedimiento es relativamente sencillo (ver figura 5.1):

- En primer lugar es necesario saber el compás que el usuario desea para la canción.
- A continuación se selecciona la primera pista del individuo en cuestión y se analizan tantos bits como indique el segundo número que forma el compás, es decir, si el compás que desea el usuario para la composición es 2/4, se seleccionarán los primeros 4 bits de cada pista.
- Una vez seleccionados esos valores, se compara esa subcadena binaria obtenida con el resto de subcadenas de la misma longitud que forman la pista.
- Por cada aparición de esa subcadena en la pista se otorga una puntuación. El valor total de esa pista será el resultado de sumar la puntuación necesaria por cada aparición.
- Este proceso se repetirá hasta que se complete la exploración de todas las pistas de cada individuo, obteniéndose finalmente un valor del fitness para cada individuo, que será la suma de la puntuación de todas las pistas que lo compongan.

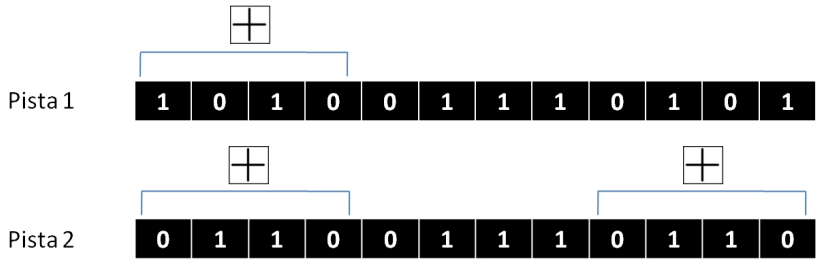

Figura 5.1: Fitness basado en patrones aplicado a dos pistas de un individuo

Como se puede observar en la figura y siguiendo con el ejemplo del compás 2/4, se cogerían las 4 primeras posiciones de la pista y se comprobaría si ese patrón aparece en algún otro lugar de la misma. En el primer caso se puede ver como no existe un patrón igual al buscado, pero en el segundo aparece repetido en una ocasión, de manera que se le otorgaría una puntuación positiva a esa pista y por tanto a la composición.

De esta manera el resultado, tras varias ejecuciones del algoritmo genético, será el de canciones que repitan de manera uniforme y con una velocidad establecida mediante la longitud de la pista y la duración de la canción elegida por el usuario, un mismo patrón en cada pista. Nótese que la repetición de patrones es únicamente por pista y no en toda la canción, ya que de esta manera se consigue que cada una de ellas reproduzca un patrón diferente que al reproducirse simultáneamente origina una composición con muchas posibles variaciones.

5.1.2. Fitness basado en patrones principales y patrones secundarios

Esta función se encarga de dividir el array binario que forma cada una de las pistas del individuo para buscar lo que se han llamado patrones principales y patrones secundarios o compatibles.

Los patrones principales son aquellos que cumplen las condiciones que se detallaban en la sección anterior, es decir, aquellos que tienen la longitud indicada por el segundo número que forma el compás. Los patrones secundarios son patrones que el sistema reconocerá como compatibles con respecto al principal. La forma de saber si un compás es compatible o no con otro es estudiando nuevamente el segundo número del compás. Si los números son divisibles o múltiplos uno del otro y se encuentran en el sistema como compases aceptables, se considerarán compatibles y serán estudiados al igual que en el caso de los patrones principales.

Por tanto el funcionamiento de esta función de fitness es el siguiente (ver figura 5.2):

- Como en el caso anterior, se obtiene el valor del compás elegido por el usuario.
- Se procede como en el caso anterior calculando y sumando una determinada puntuación cada vez que una de las subcadenas resultantes coincidan con el patrón seleccionado.

- Una vez se ha obtenido el valor del fitness principal para un individuo, pasan a calcularse cuales son los compases compatibles con el existente. Para ello se calcula cuales de los compases posibles son múltiplos o divisibles del número en cuestión.
- Por cada uno de ellos se repite el proceso anterior, pero tomando como patrón tantos bits como indique el segundo número del compás compatible, es decir, si un compás compatible con el compás 2/4 resulta ser el 1/2, se estudiará nuevamente la pista buscando patrones que coincidan con los 2 primeros bits de la pista.
- Cuando se encuentre una subcadena igual al valor del patrón buscado, se sumará una determinada puntuación que será siempre menor a la correspondiente al fitness principal.
- Este proceso se repetirá hasta que se complete la exploración de todas las pistas de cada individuo, tanto con el compás principal como con todos los secundarios, obteniéndose finalmente un valor del fitness para cada individuo que será la suma de la puntuación del fitness primario y secundario de todas las pistas que lo compongan.

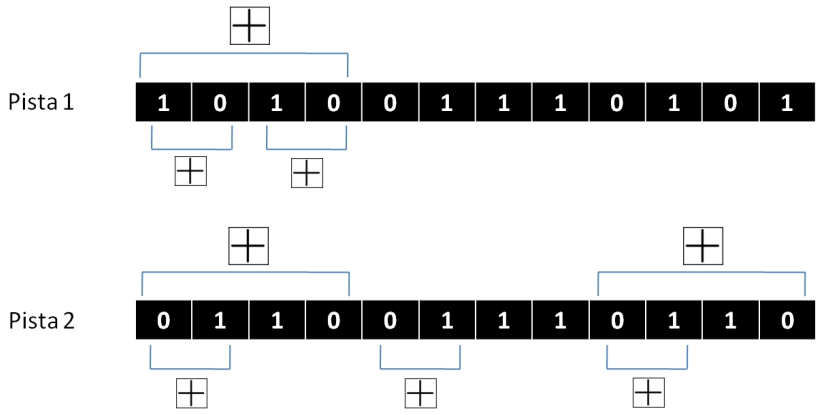

Figura 5.2: Fitness basado en patrones principales y secundarios aplicados a dos pistas de un individuo

Siguiendo con el ejemplo anterior, se puede comprobar como en este caso se ha añadido un fitness compatible al 2/4, como es en este casi el 1/2. En la parte inferior de cada pista se ha realizado una división de la misma en porciones de dos notas, de manera que se recorre el array completo buscando

coincidencias con el correspondiente a las dos primeras posiciones del propio array. En el primer caso la cadena "10" aparece dos veces mientras que en el segundo, la cadena buscada "01" aparece en tres ocasiones.

Añadiendo esta nueva característica se consigue aumentar la variabilidad de los resultados y en función de la diferencia existente entre la puntuación otorgada al principal y al secundario, se le puede dar más peso a un factor u otro según sean los resultados perseguidos. Además, otra posibilidad es la de hacer disminuir el valor de la puntuación otorgada en función de la distancia al número estudiado, es decir, que si se continúa con el ejemplo de encontrarse frente a un compás 2/4, se otorgue más peso al compás compatible 1/2 que al 8/16 debido a la distancia existente entre uno y otro.

5.1.3. Fitness basado en patrones y ritmos

En este enfoque entra de una forma más plausible el concepto del ritmo. Se añade el carácter rítmico al intentar asegurar un golpe de sonido en el lugar en el que se debe marcar el ritmo.

Para lograr este objetivo se tiene en cuenta el primer número que forma el compás y se puntúa de forma positiva si en aquellas posiciones de la pista que se corresponden con las posiciones donde debe empezar el compás (en cualquier pieza se produce un golpe más fuerte en esos momentos), se reproduce un sonido, o lo que es lo mismo, existe un "1" en el array binario de notas.

El procedimiento para calcular este fitness es similar al de los casos anteriores (ver figura 5.3):

- Se calcula el fitness primario y secundario de las pistas como se ha indicado en los apartados anteriores.
- Posteriormente, tras seleccionar el primer número que forma el compás, se selecciona la primera pista del primer individuo y se recorre comprobando si en las posiciones del array de notas que son múltiplos del número correspondiente del compás existe un "1". Si es así, se suma un determinado valor a esa pista y en caso contrario no se realiza acción alguna.
- Este proceso se repetirá por cada pista del individuo y el valor del fitness del individuo será la suma de este fitness junto con el primario y el secundario, obtenidos en los pasos anteriores.

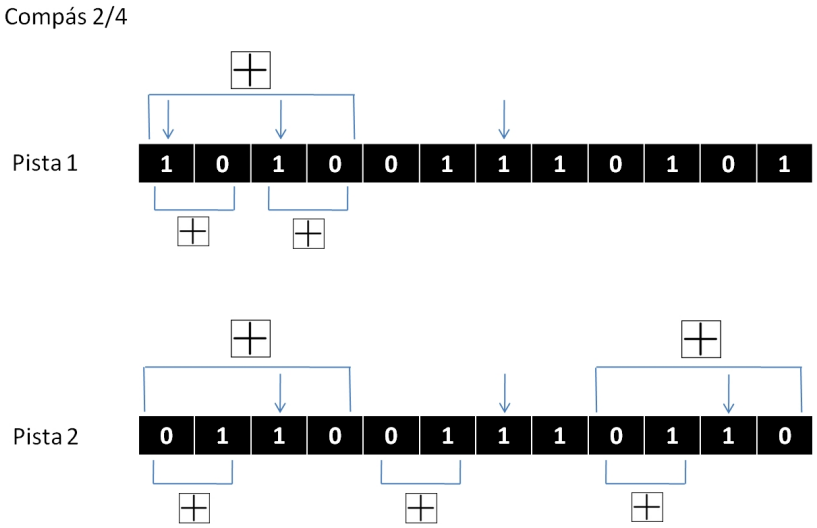

Figura 5.3: Fitness basado en patrones y ritmos aplicado a dos pistas de un individuo

En la figura 5.3 se representan con una flecha aquellas posiciones que cumplen lo mencionado arriba y son puntuadas de forma positiva. Se puede comprobar que aquellas posiciones donde se debe marcar el compás y aparece un "0" no poseen flecha y no reciben puntuación positiva o negativa.

Este tipo de fitness ayuda a que la pieza resultante contenga más información musical y además permita representar claramente el ritmo y el compás, estableciendo una diferenciación entre, por ejemplo, un compás 2/4 y un 3/4, situación que no podía darse en las anteriores versiones.

5.1.4. Fitness vertical

Con esta función de fitness se pretende que las piezas que se obtengan no acumulen demasiados sonidos en un mismo instante de tiempo con el fin de evitar la confusión y el "ruido" durante la reproducción de la base rítmica.

Para su desarrollo se han estudiado las diferentes pistas que componen la base rítmica en vertical, es decir, la primera nota de todas las pistas en primer lugar, la segunda de cada una de ellas después y así sucesivamente. Si el número de unos que se encuentren en cada una de esas agrupaciones supera un determinado umbral, esa composición será puntuada negativamente, mientras que por cada vez que no se produzca esa situación se puntuará de forma positiva.

En este fitness también se ha tenido en cuenta el compás que sigue la composición. Como se ha comentado en secciones anteriores, el ritmo se marca con un golpe más acusado de los instrumentos que se estén reproduciendo. Para ser consecuente con ese convenio y permitir a las composiciones resultantes obtener ambos objetivos de forma conjunta, este fitness sólo será aplicado en aquellas posiciones del array de notas que no se correspondan con aquellos en los que deba aparecer el golpe del compás. De esta manera, en las posiciones que se deba marcar el compás podrán sonar a la vez tantos sonidos como pistas haya, pero en el resto sólo un número inferior al umbral establecido.

Por lo tanto, el proceso a seguir para calcular el valor de esta función de fitness es el siguiente (ver figura 5.4):

- Se calcula la puntuación asociada a los fitness detallados anteriormente.
- A continuación, se selecciona la primera posición de cada una de las pistas que compongan la canción.
    - Si esa posición se corresponde con una de las posiciones donde se marca el compás, se selecciona la siguiente posición para todas las pistas.
    - Si no se corresponde con una posición en la que se debe marcar el compás, se cuenta el número de unos que aparecen.
        - Si supera el valor umbral establecido para esa composición, se puntúa negativamente a la misma.
        - Si no supera el valor establecido, se otorga una puntuación positiva a la composición.
- Este proceso se repetirá por cada posición de las pistas existentes hasta completar la total longitud de las mismas, siendo el valor de este fitness la suma resultante de la puntuación otorgada a cada una de las posiciones de las pistas, y pudiendo ser esta positiva o negativa.

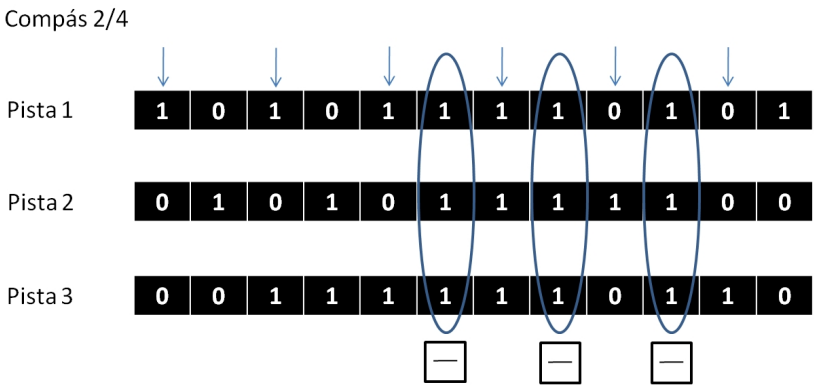

Figura 5.4: Fitness basado en el estudio vertical de las pistas

Como se puede observar en la figura 5.4, aparecen tres pistas correspondientes a una composición. El compás de la misma es 2/4 y las flechas indican las posiciones en las que se marca el compás. Es fácil comprobar como aquellas posiciones que se encuentran encerradas en la elipse están formadas íntegramente por unos, de manera que se superaría el valor umbral y se le otorgaría una puntuación negativa. También se puede comprobar como se produce esta situación en otra posición correspondiente a un golpe del compás, pero en este caso no se aplica debido a las razones antes comentadas.

Con esta nueva función de fitness se consigue que la base rítmica sea más agradable gracias a la limitación de sonidos en un mismo momento que pueden hacer la composición pesada y "ruidosa". Además mediante este fitness se consigue establecer aún mejor la diferencia entre los momentos donde se debe marcar el compás y los que no, otorgando más orden y ritmo a la base rítmica.

5.1.5. Fitness orientado al bajo

Esta función de fitness se aplicará únicamente a aquellas pistas que reproduzcan sonidos pertenecientes al bajo. Su misión es la de conseguir que el bajo y la batería se reproduzcan de forma coordinada en la canción.

Un hecho constatado es que en la mayoría de las piezas musicales, el sonido que produce un bajo acompaña siempre al bombo de la batería. Éste será el funcionamiento de esta función de fitness. Se encargará de puntuar positivamente cuando el sonido del bajo coincida con un sonido del bombo y no se otorgará ninguna puntuación en caso contrario.

Si se produce el hecho de que en una canción no se desea introducir un bombo de batería y sí un bajo, esta función de fitness no se aplicará a ninguna de las pistas.

A continuación se explican los pasos que han de seguirse para obtener el valor de fitness asociado a esta función:

- En primer lugar se busca en las diferentes pistas que componen la canción si alguna de ellas lleva asociada el sonido del bombo.
  - Si no existe ninguna, se sale de la función
  - En caso afirmativo, se almacena como el modelo que deberá seguir el bajo.
    - Una vez almacenado el modelo que se seguirá, se comprueba si alguna de las otras pistas de la canción llevan un sonido del bajo asociado. Si no es así, no se aplicará la función de fitness, pero en caso de que exista se almacenará el valor de la pista.
    - Una vez almacenados los valores de las dos pistas, empezarán a compararse posición por posición. Cada vez que coincida el valor de la misma posición en las dos pistas se sumará una determinada puntuación. En caso de que no coincidan no se realizará acción alguna.
    - Cuando se ha completado este proceso se sumará el valor del fitness de cada pista así como el de las funciones de fitness explicadas en los puntos anteriores, obteniéndose de esta manera el fitness total de la canción.

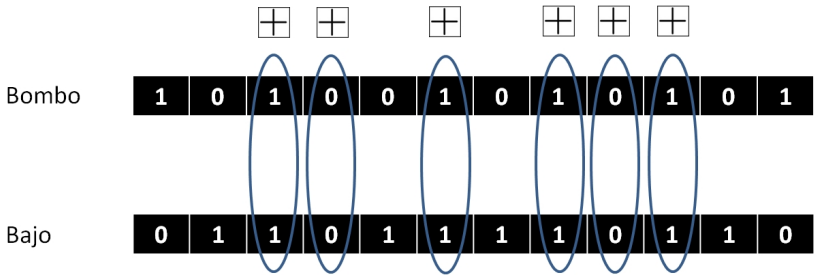

Figura 5.5: Fitness aplicado al instrumento del bajo

5.2. Sistema de creación interactivo

Como se comentaba al inicio de este capítulo, la versión interactiva del sistema hace necesaria la participación activa del usuario en el proceso de composición. Ésa es la principal diferencia con el sistema automático, pero no la única, ya que algunos elementos más son modificados para adaptarse a las exigencias derivadas de este tipo de evaluación.

La primera modificación clave es la posibilidad de que el usuario otorgue una puntuación o valoración personal a cada una de las composiciones que escuche. Esta puntuación influirá en la evolución de las canciones generadas, ya que en función de lo alta o baja que sea la puntuación del usuario, se multiplicará el valor del fitness de esa composición por un valor alto, que hará que las características del individuo se sigan propagando por la población, o por un valor más bajo, que hará descender el fitness del individuo en cuestión aumentando las posibilidades de que desaparezca y sea sustituido por un individuo más adecuado para el usuario.

Para hacer más sencilla la labor del usuario y a la vez más interesantes los experimentos realizados, se hacía necesario evitar que las canciones que se propusieran para puntuar se pareciesen demasiado entre sí. Para solucionar esté problema se pensó utilizar la técnica de nichos o crowding, que consiste en potenciar la diversidad genética de los mejores, es decir, no sólo buscar evolucionar obteniendo los mejores individuos sino con el objetivo de obtener los mejores más distintos.

Cuando se comenzó a desarrollar esta técnica se observó que, tras gran cantidad de iteraciones, las canciones presentaban aún bastantes similitudes entre ellas. De esta manera se pensó en emplear otra técnica que permitiría dejar atrás este obstáculo con bastante seguridad, el modelo de islas.

Esta idea se basa en la que Darwin plasmó en su obra [Darwin, 1845]. Cuando Darwin llegó a las Islas Galápagos en 1835 encontró una notable variedad de pájaros pinzones que supusieron un caso muy atractivo y sugerente para el desarrollo de su teoría de la evolución. El archipiélago de las Islas Galápagos es un conjunto de 29 islas e islotes de diferente tamaño a 600 millas de la costa de Ecuador. Los pinzones se alimentaban habitualmente de semillas en el suelo y poseían picos gruesos para romper las duras cubiertas de las semillas. Las especies de Las Galápagos, aunque pinzones todos ellos, muestran una inmensa variación en su forma de vida, la forma de sus picos y su conducta, bajo las cuáles subyacen importantes diferencias ecológicas. Esta diversidad de especies surgió a partir de una población original de comedores de semillas que llegó a Las Galápagos procedentes del continente americano. Los descendientes de los colonizadores iniciales se dis-

tribuyeron por las distintas islas y en zonas diferentes, formando poblaciones locales que fueron diferenciándose unas de otras y creando ocasionalmente especies distintas. Pero, como cualquier población de cualquier especie, el tamaño de estas poblaciones es finito, por lo que la evolución podría quedar estancada si no se producen variaciones. Dichas variaciones, además de por mutación, se obtienen en este caso por migración, que consiste en que cada cierto tiempo un individuo de una de las poblaciones pasa a formar parte de otra, consiguiendo la variación deseada.

Aplicando esta idea al entorno que nos concierne, se puede plantear la situación de forma que al lanzar el sistema se creen varias poblaciones sobre las que se irá ejecutando el algoritmo genético en paralelo, obteniendo tantas líneas o islas de evolución diferente como poblaciones hayan sido creadas y consiguiendo el objetivo de obtener varias composiciones diferentes entre sí para que el usuario decida cual es más de su agrado.

Dado que al usuario se le expondrán seis canciones diferentes, el sistema empleará tres poblaciones o islas para realizar la evolución de las mismas en paralelo. De esta manera las seis composiciones se distribuirán como sigue:

1. En el primer botón aparecerá el individuo de mayor fitness de la primera isla.

2. El segundo botón se corresponderá con el mejor individuo de la segunda isla.

3. El tercero, con el mejor individuo de la tercera población.

4. El cuarto saldrá de hacer un cruce entre el mejor de la primera población con el mejor de la segunda.

5. El quinto será el resultado de realizar un cruce entre el mejor individuo de la segunda población y el mejor de la tercera.

6. El sexto y último corresponderá al individuo resultante del cruce del mejor de la primera población con el mejor de la tercera.

Una vez establecidas las correspondencias entre los botones y la composición que se reproducirá, sólo queda definir qué tipo de cruce se aplicará entre los individuos de las diferentes islas.

Se plantearon diversas opciones en este aspecto, todas ellas interesantes pero que en algunos casos no cumplían algunos de los requisitos deseados. Finalmente se decidió que el cruce entre los mejores se realizaría por cromosoma, es decir, se intercambiaría una pista de uno de los individuos con otra

pista cualquiera del individuo de la isla correspondiente, ambas escogidas de manera aleatoria. De esta manera se conseguían objetivos planteados como el de evitar el estancamiento o la posibilidad de obtener composiciones más diferenciadas entre sí, facilitando la labor del usuario. Además el hecho de realizar ese tipo de cruce da lugar a nuevas combinaciones entre pistas que pueden desembocar en ritmos originales y diferentes, todo ello asegurando siempre un cierto orden debido al hecho de que la pista intercambiada habrá sido evolucionada (en mayor o menor medida) con anterioridad.

Con estas características añadidas se presenta el modo interactivo del sistema. Merece la pena destacar que el resto de aspectos se mantendrán de la misma manera que se hacía en el caso del modo automático y las funciones de fitness que se aplicarán serán exactamente las mismas que en el caso anterior. La diferencia radica en que esta vez se ejecutará cada paso del algoritmo tantas veces como islas o poblaciones haya (en este caso tres) pero proporcionando siempre resultados y configuraciones independientes.

En cuanto a la pruebas, comentar que su realización y explicación en este documento carece de relevancia e interés, debido a que esta versión del sistema se encuentra diseñada para construir composiciones según los deseos o gustos de un usuario en concreto, haciendo imposible distinguir esos resultados como buenos o malos desde un punto de vista musical. Es la propia subjetividad implícita en ese proceso la que hace imposible su evaluación. Por tanto, se han realizado los experimentos pertinentes para comprobar el correcto funcionamiento del sistema y su adecuación a diversas configuraciones y situaciones, pero no se han realizado aquellas encaminadas a evaluar el rendimiento y los resultados obtenidos, por lo que no se mostrará prueba alguna en el capítulo siguiente.

Capítulo 6

Experimentación

En este proyecto uno de los núcleos y valores más importantes reside en la experimentación. Es importante recalcar una vez más el carácter innovador de este sistema y los retos y dificultades que se afrontan. Por todo ello esta sección posee una especial relevancia. Los experimentos aquí desarrollados y estudiados arrojarán resultados que pueden servir de base para investigaciones posteriores y permitirán la extracción de una serie de conclusiones siempre interesantes.

En primer lugar es necesario definir los requisitos que deben de ser cumplidos por los experimentos para que los resultados obtenidos puedan ser extraídos y estudiados de forma coherente y siguiendo las características de la computación evolutiva en general y de este proyecto en particular. Los requisitos a completar son los siguientes:

- Las pruebas desarrolladas deben permitir la visualización de los resultados en cualquier momento deseado, es decir, debe generar resultados no volátiles.
- Se debe permitir realizar tantas pruebas como se desee.
- Una prueba no debe sobreescribir los resultados de la anterior.
- Las pruebas estarán encaminadas a la comprobación de la evolución de piezas musicales para su estudio posterior, siendo necesario que la información obtenida sea almacenada de forma ordenada y razonable.
- Se deberá generar información no volátil acerca de qué parámetros de ejecución se han tomado al ejecutar una determinada prueba.

- Las pruebas se deberán implementar de forma que se garantice que sus resultados no son debidos al azar.

Una vez conocidos los principales requisitos que deben cumplir los experimentos a realizar se hace necesario establecer una serie de métodos y procedimientos que permitan realizar cada una de las pruebas y visualizar los resultados de forma lógica y coherente. Para este propósito se ha desarrollado una parte independiente al sistema que permitirá la ejecución de pruebas con los parámetros de configuración deseados de forma automática, existiendo la posibilidad de ejecutar varios experimentos seguidos sin necesidad de dar la orden al programa. Además, todos los resultados obtenidos de las diversas pruebas quedarán almacenados en ficheros de texto que podrán ser consultados y estudiados cuando se desee. Los parámetros a configurar en este programa son los siguientes:

- Longitud de la canción
- Número de pistas
- Número de instrumentos
- Longitud de cada pista
- Compás
- Instrumentos que se desean añadir
- Número de pruebas a ejecutar
- Número de individuos que tendrá la población a ejecutar
- Número de generaciones que se aplicarán en la ejecución del algoritmo genético
- Porcentaje de mutación a aplicar en el algoritmo genético

Además será necesario especificar cuál o cuáles de las funciones de fitness desarrolladas se emplearán, ya que los resultados pueden variar de unas a otras.

Con la indicación por parte del usuario de todos estos parámetros, el programa lanzará las pruebas necesarias con esa información y los resultados quedarán recogidos en los ficheros de texto para su análisis.

Tras establecer las bases sobre las que se asentarán las pruebas a realizar, se deberá organizar la forma y orden en el que éstas serán ejecutadas. El plan que se seguirá para la ejecución de las pruebas será el siguiente:

1. Establecer el valor óptimo para cada uno de los operadores genéticos de forma que se maximicen los resultados obtenidos, creando la mejor configuración posible.

2. Establecer el valor que se otorgará a cada una de las partes que forman el fitness para que los individuos resultantes se acerquen lo máximo posible a los ideales establecidos.

3. Realizar una batería de pruebas y seleccionar las más interesantes para que sean detalladas y explicadas en este documento.

4. Se repetirá el mismo conjunto de pruebas para todas las funciones de fitness comentadas en el capítulo anterior con el fin de estudiar y comparar las salidas obtenidas, estableciendo las conclusiones oportunas en cada caso.

Como información adicional, aclarar que todos los gráficos que se muestren a lo largo del capítulo reflejarán en el eje de abscisas el número de generaciones que se lleven ejecutadas y en el eje de ordenadas el valor del fitness ponderado de 0 a 100.

Una vez definidos los pasos que se deben seguir para realizar una buena fase de experimentación, se comienza a desarrollar el plan comentado anteriormente y mediante el cual se examinará cada uno de los aspectos del proyecto obteniendo las conclusiones oportunas en cada caso.

6.1. Operadores genéticos

El primer punto del sistema a definir debe ser cuál de los operadores genéticos implementados serán aplicados y los motivos que llevan a ello. Para tomar la decisión de la forma más adecuada, se estudiará cada uno de ellos de forma independiente y se elegirá el que mejores resultados obtenga o el que mejor se adapte a la situación en cuestión. El número de experimentos y configuraciones probadas es muy elevado por lo que se hará una selección de las pruebas y resultados más interesantes para que sean mostrados en este documento.

Un dato importante y determinante en el resultado de las pruebas de los operadores genéticos es que se empleará la función de fitness de patrones principales y secundarios. Se ha elegido esta función para la elección de los operadores por ser la función base sobre la que se asientan el resto de funciones y por considerar estos los aspectos más importantes del sistema, por ser los encargados de localizar y premiar aquellos patrones que pueden

resultar más rítmicos y beneficiosos y que supondrán la base de creación de la canción. En lugar de ésta, e podría haber utilizado la función que únicamente busca los patrones principales, pero se ha considerado que la elegida se acerca más a la realidad perseguida y no es tan general y sencilla como la anterior.

6.1.1. Selección

La selección es el proceso mediante el cual se decide qué individuos de una población pasarán a formar parte de la siguiente. Esta decisión se toma en base a un proceso de selección que varia según el tipo de implementación elegido.

En este proyecto se ha decidido emplear la selección por torneo. La elección de la selección por torneo viene determinada por el deseo de aumentar al máximo las posibilidades de que perduren los mejores individuos generación tras generación. En primer lugar se utilizó un torneo de 3 individuos pero posteriormente se aumentó hasta 5 con la idea de incrementar las probabilidades de alcanzar ese objetivo. Por ello, lo largo de todas las pruebas se empleará una selección por torneo de 5 individuos, número que se pensó más adecuado debido a que hacía disminuir la probabilidad de perder las mejores muestras y permitiendo aún así obtener la suficiente variedad genética.

A pesar de ello, las primeras pruebas realizadas mostraron la dificultad de evolución de la población en determinadas circunstancias y tras un estudio detallado se pudo comprobar como en la gran mayoría de ellos se debía a la perdida de los mejores individuos que hacían retrasar la evolución favorable de la población. Como medida para solucionar este problema se decidió emplear *elitismo*.

El *elitismo* se trata de una técnica mediante la cual se asegura la permanencia de una generación a otra del mejor o mejores individuos de la población sin que sufran variación alguna, es decir, sin que pasen por las fases de selección, cruce y mutación. Gracias a este método se consigue que la tendencia del mejor fitness sea ascendente o como mínimo se mantenga, permitiendo en la gran mayoría de los casos que el sistema evolucione de una forma más rápida mediante la eliminación del riesgo de perder el mejor individuo.

Los resultados de las pruebas realizadas entonces fueron mucho más adecuados, observando una clara evolución positiva de las poblaciones y además de forma mucho más rápida, hecho que apoyó las decisiones tomadas.

6.1.2. Cruce

El cruce es el proceso mediante el cual dos individuos intercambian su código genético. Como bien es sabido, es el proceso equivalente a la reproducción sexual de los mamíferos.

Existen diversas formas de implementar el cruce y todas ellas son válidas, aunque unas opciones se adaptan mejor a unas situaciones que a otras. Como se comentó en el capítulo anterior, se han implementado tres tipos de cruce diferentes y se han realizado las pruebas correspondientes con cada uno de ellos con el fin de comprobar cual de ellas funciona mejor en el contexto en el que se está trabajando.

Se cruzarán todos los individuos de la población salvo el mejor de la generación anterior que, como se comentaba y debido al elitismo, se encuentra exento de pasar por los diferentes operadores genéticos. Para asegurar que todos los individuos sean cruzados se ha procedido a realizar el cruce siempre entre individuos consecutivos de la población cogidos de dos en dos empezando por el segundo. Si se trata de un número par de individuos (no se debe olvidar el extraído por el elitismo) se dejará sin cruzar el último individuo de la población.

A continuación se realizarán una serie de pruebas que permitirán determinar el tipo de cruce más adecuado para las pruebas sucesivas. Para todas ellas los parámetros de configuración de la canción y del algoritmo genéticos son los mismos:

- 101 individuos
- 3000 generaciones
- 3 pistas por canción
- 100 notas por pista
- Compás 2/4
- Selección por torneo de 5 individuos
- Porcentaje de mutación de 0,01

Los resultados obtenidos tras estas pruebas quedan reflejados en las gráficas que se presentan seguidamente. Todas ellas están ponderadas, siendo 100 el máximo que se puede alcanzar:

1. En primer lugar se presenta la figura 6.1 correspondiente a la evolución del cruce simple.

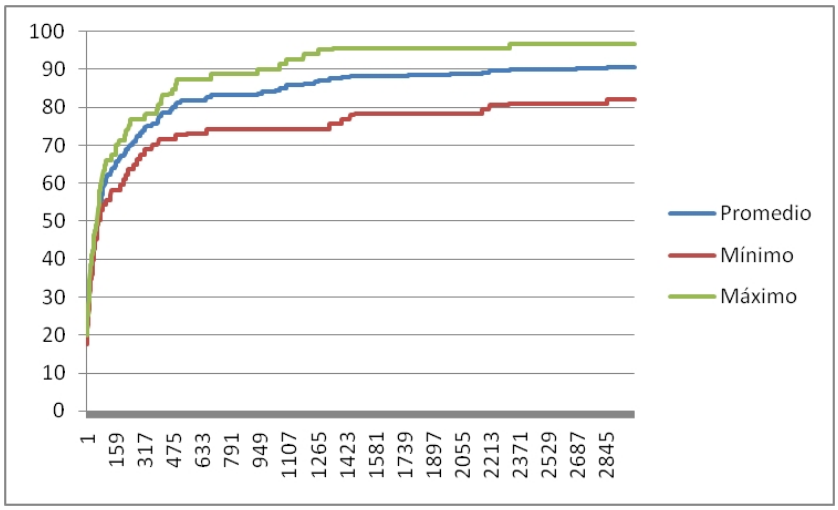

Figura 6.1: Evolución de la población utilizando cruce simple

Como se puede comprobar en la figura 6.1, su evolución es positiva, acercándose bastante al máximo en promedio con aproximadamente un 90 %, aunque tal vez algo más lenta de lo que se desearía. Aún así, existen individuos buenos que se acercan aún más al máximo alcanzable y que proporcionarían resultados aceptables, pero la distancia existente entre los parámetros estudiados puede indicar grandes diferencias entre los individuos de la población, en las que se encontrarían individuos buenos y otros nada aceptables. También comentar que la evolución parece que aún no se ha estancado y que podría mejorar los resultados si se dejase ejecutar un número más alto de generaciones.

2. El siguiente cruce a estudiar será el multipunto.

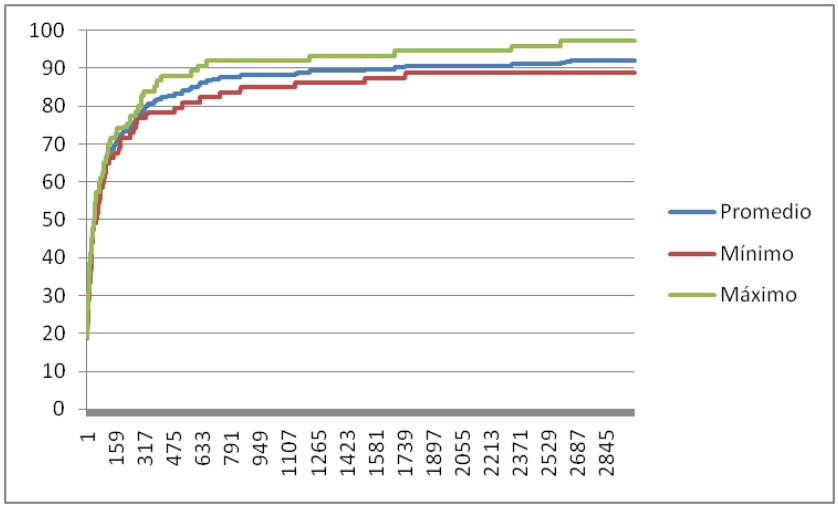

Figura 6.2: Evolución de la población utilizando cruce multipunto

Es sencillo ver en la figura 6.2como en este caso la evolución sigue una tendencia similar pero con una curva más pronunciada y obteniendo mejores resultados. También es importante remarcar el hecho de que las distancias existentes entre el promedio, el máximo y el mínimo son mucho menores que en el experimento anterior lo que indica que los individuos obtenidos finalmente son más adecuados y parejos, y no dependen tanto del hecho de encontrar un individuo muy cercano al máximo sino que es fruto de la evolución de la población completa. Como en el caso anterior, la tendencia sigue siendo positiva y es claro que dejándolo ejecutar un número algo más elevado de generaciones se podrían mejorar los resultados obtenidos.

3. Por último se estudia el funcionamiento del cruce por compás.

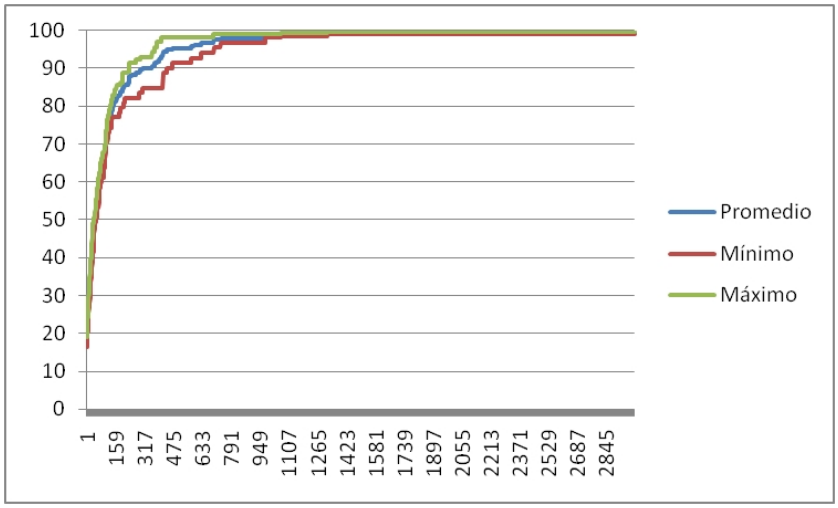

Figura 6.3: Evolución de la población utilizando cruce por compás

Las diferencias con las pruebas anteriores es clara como se puede apreciar en la figura 6.3. La evolución es muchísimo más rápida, aproximadamente en la generación 1000 ya se encuentra rozando el máximo, y alcanza cotas de resultados mucho más altas. El "problema" de este tipo de cruce es la alta probabilidad de que los individuos resultantes sean muy similares entre sí, ya que al cruzar compases enteros, en cuanto se encuentra uno de ellos que funciona bien en varios de los individuos, éste se propaga con rapidez al resto de la población. Aún con este "problema", los resultados obtenidos son muy buenos y no debe suponer una limitación debido a las múltiples opciones de creación que tiene el sistema que hacen que sea muy complicado que en distintas ejecuciones se produzcan los mismos individuos, eliminando este problema planteado.

Para ver y comparar de forma más clara los resultados obtenidos por cada uno de los tipos de cruce, se presenta la tabla resumen 6.1:

	Promedio	Mínimo	Máximo
Simple	90,4403495	82,026992	96,6649364
Multipunto	92,0667878	88,8139112	97,3008046
Compás	99,3252011	99,0007786	99,6496237

Cuadro 6.1: Tabla resumen de pruebas de operador de cruce (30 experimentos)

En ella se pueden corroborar las afirmaciones realizadas y observar como el cruce por compás es bastante más eficiente que los otros, alcanzando un valor de fitness promedio superior al 99 %. Por esta razón se empleará el cruce por compás en el resto de pruebas, aunque en algún momento se mostrará cómo funcionan el cruce simple y el multipunto bajo otro tipo de configuración.

6.1.3. Mutación

La mutación es el proceso por el cual se modifican determinados genes de los individuos de forma aleatoria con el objetivo de evitar el estancamiento y favorecer la diversidad genética.

Como bien es sabido, la mutación es un elemento determinante en un algoritmo genético y asegura en cierto modo, la evolución de las poblaciones. Con estos datos presentes, se podrá comprobar como el efecto de la mutación en este sistema no es el que cabía esperar en un primer momento.

Se han implementado dos tipos de mutación. El primero es la clásica mutación que se encarga de alterar un determinado número de bits de forma aleatoria, mientras que el segundo tipo se encarga de seleccionar un compás y cambiarlo por otro generado aleatoriamente en ese instante.

A continuación se procederá a realizar una serie de pruebas con la siguiente configuración en la que se irá variando tanto el porcentaje como el tipo de mutación utilizado:

- 101 individuos
- 3000 generaciones
- 3 pistas por canción
- 100 notas por pista
- Compás 2/4

- Selección por torneo de 5 individuos
- Cruce por compás

Ahora se presentarán una serie de pruebas y gráficos, del amplio banco realizado, que muestran el comportamiento de la mutación en el sistema:

1. Mutación clásica

   Como se ha comentado la mutación clásica se encarga de modificar genes del individuo de forma que si un valor obtenido de manera aleatoria se encuentra por debajo de un umbral establecido se modifica y en caso contrario se deja como está.

   A continuación se presentan distintos valores umbrales para los que se han realizado pruebas y los resultados obtenidos en cada una de ellas:

   - Mutación 0,01

     Esta mutación se corresponde con un 1 % de la población si se habla en términos estadísticos. Es una mutación pequeña y la más común en este tipo de problemas. En la gráfica 6.4 se pueden comprobar los resultados utilizando este porcentaje de mutación.

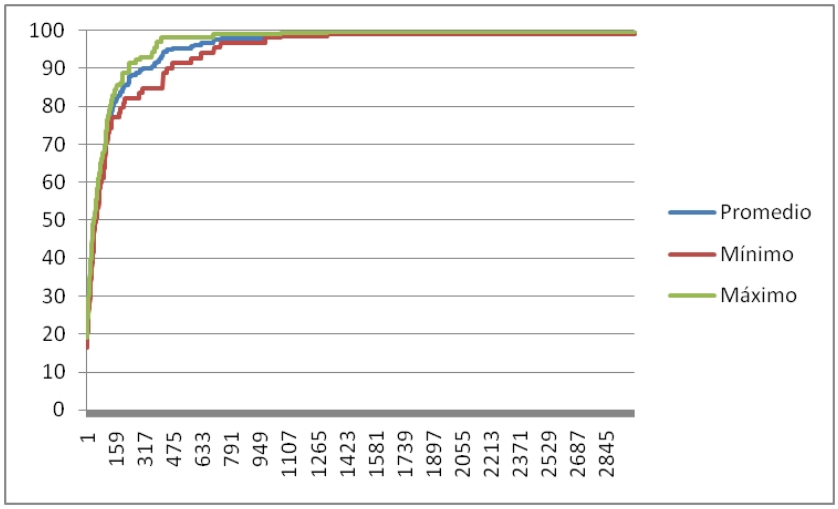

Figura 6.4: Evolución de la población utilizando mutación clásica de 0,01

Los resultados obtenidos son muy buenos ya que la población evoluciona de forma favorable y rápida, llegando a alcanzar valores

muy cercanos a los máximos.

- Mutación 0,02
  Para esta prueba se decide aumentar algo el porcentaje de mutación y así observar que diferencias se producen con el caso anterior. El gráfico 6.5 representa la evolución de este sistema bajo la influencia de este parámetro.

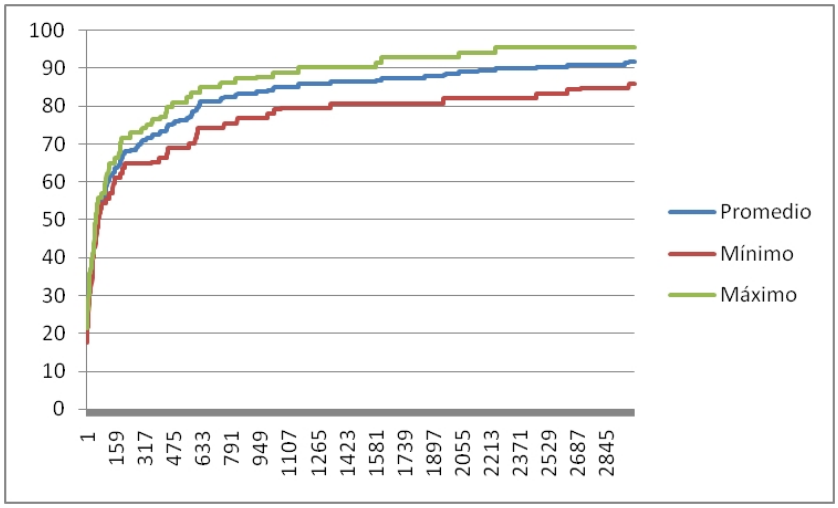

Figura 6.5: Evolución de la población utilizando mutación clásica de 0,02

Como se puede ver, en este caso la evolución del sistema es mucho más lenta, y aunque parece que sigue una tendencia positiva y con posibilidad de mejorar, queda reflejada la diferencia existente con el caso anterior.

- Mutación 0
  Con esta prueba se pretende comprobar la reacción del sistema sin la existencia de mutación y poder enfrentar los experimentos realizados hasta ahora.

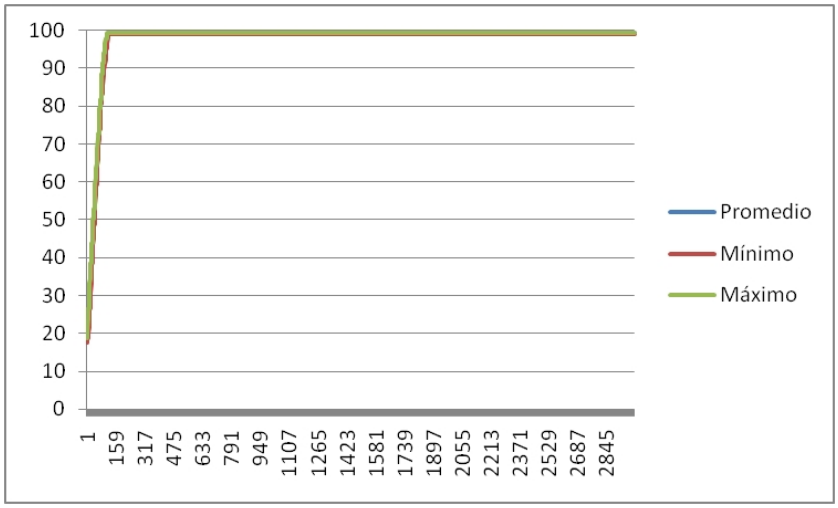

Figura 6.6: Evolución de la población sin utilizar mutación

Los resultados arrojados por esta configuración son sorprendentemente positivos. Aproximadamente en la generación 200 alcanza valores muy cercanos al máximo. La evolución por tanto es muy rápida y favorable a los intereses de la función de fitness empleada. Experimentos posteriores nos permitirán extraer unas conclusiones más razonadas y claras de este suceso en el que la mutación está afectando negativamente al algoritmo genético.

2. Mutación por compás

   La mutación por compás se encarga de seleccionar un gen de manera aleatoria, calcular el compás al que pertenece y sustituirlo por otro generado de forma aleatoria en ese instante. En este caso el número de bits seleccionados para mutar será bastante más reducido ya que por cada uno de ellos que se seleccione se modificarán tantos como indique el compás de la canción. Para ello se ha desarrollado una fórmula para calcular que el número bits a cambiar sea equivalente a los de la mutación clásica.

   Seguidamente se muestran los porcentajes de mutación empleados, que serán exactamente iguales a los utilizados en la mutación clásica para poder después establecer las comparaciones y conclusiones oportunas.

   - Mutación 0,01

Los resultados tras utilizar este porcentaje de mutación se pueden apreciar en la figura 6.7:

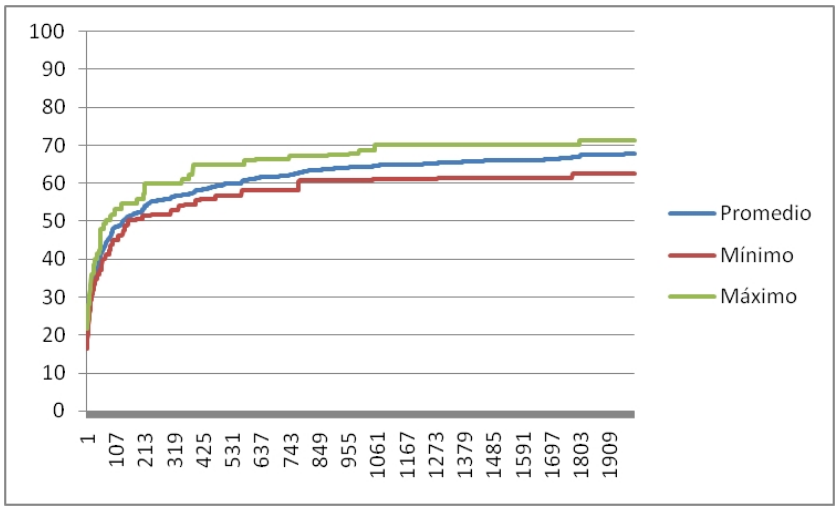

Figura 6.7: Evolución de la población utilizando mutación por compás de 0,01

En este caso se puede comprobar como la evolución es considerablemente más lenta y los resultados obtenidos se encuentran también por debajo de los del caso anterior, alcanzando tras 2000 valores cercanos al 70 %.

- Mutación 0,02

  En la figura 6.8 se muestra la evolución del sistema empleando este porcentaje de mutación:

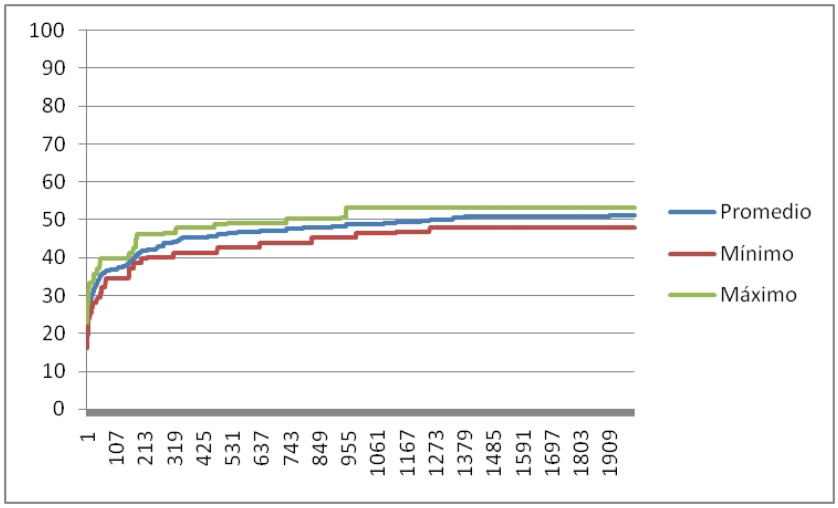

Figura 6.8: Evolución de la población utilizando mutación por compás de 0,02

Este experimento corrobora lo extraído en la mutación clásica que indica que este porcentaje de mutación no es el más adecuado, debido a que la evolución es muy escalonada y los mejores resultados sólo consiguen alcanzar el 50 % del total.

- Mutación 0

  Con el objetivo de compararlo con el caso anterior se repite el experimento sin mutación:

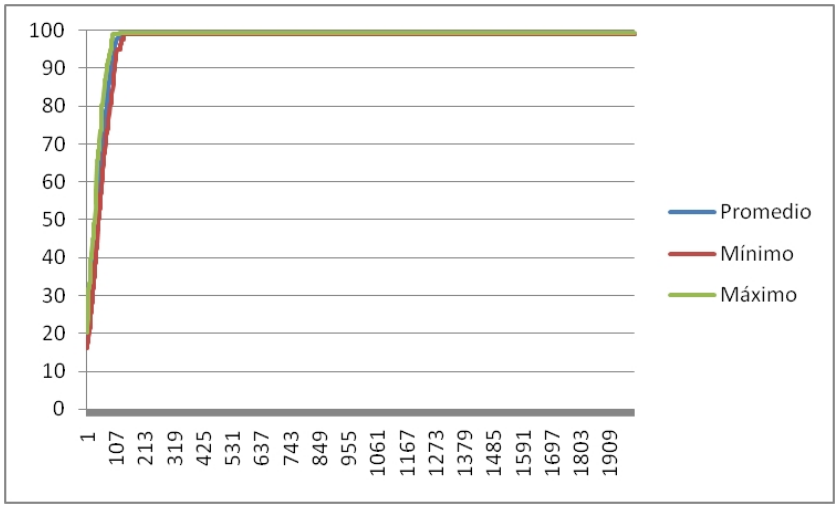

Figura 6.9: Evolución de la población sin utilizar mutación

Evidentemente, como muestra la figura 6.9 los resultados obtenidos son los mismos y demuestra la gran diferencia existente para este caso la existencia o no de mutación.

La tabla 6.2 resume los resultados obtenidos de cada una de las pruebas presentadas y permite comparar fácilmente cada una de ellas:

	Promedio	Mínimo	Máximo
Mutación Clásica – 0,0	99,2170603	99,0007786	99,3252011
Mutación Clásica – 0,01	99,3252011	99,0007786	99,6496237
Mutación Clásica – 0,02	91,8937624	85,9460161	95,6267843
Mutación Compás – 0,0	99,2170603	99,0007786	99,3252011
Mutación Compás – 0,01	67,7809499	62,6265248	71,1912795
Mutación Compás – 0,02	51,0537244	47,897742	53,153387

Cuadro 6.2: Tabla resumen de pruebas del operador de mutación

Se puede observar como la mutación clásica se encuentra claramente por encima de la mutación por compás y como su funcionamiento en este contexto es indiscutiblemente superior. Continúa llamando la atención el hecho de que sin mutación los resultados obtenidos por el sistema sean tan buenos y por ellos se realizará a continuación una serie de experimentos para determinar las posibles causas de este hecho y su rendimiento bajo otras condiciones.

La configuración que se probará a continuación será la misma que la anterior pero se aumentará el número de notas por pista de 100 a 1000 con el fin de comprobar si la no existencia de mutación ante gran número de datos puede ser perjudicial para el sistema o si lleva al estancamiento.

- Mutación 0,01

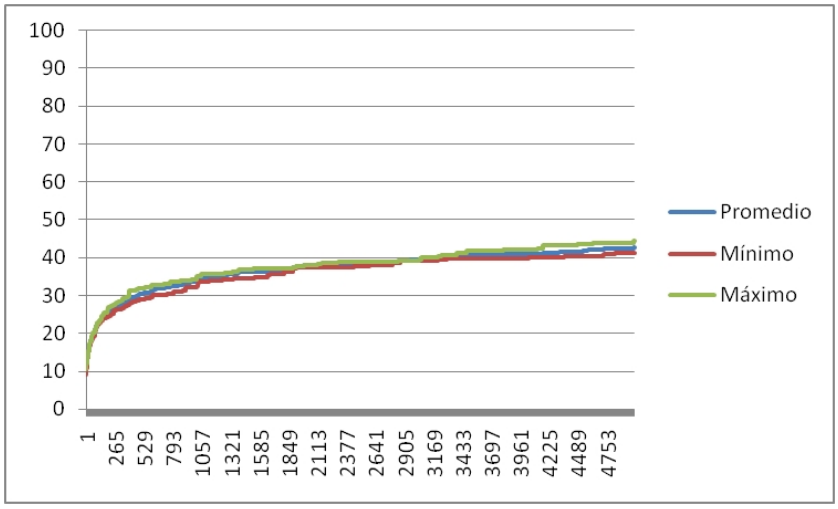

Figura 6.10: Evolución de la población con mutación clásica de 0,01

En la figura 6.10 se observa con claridad como la evolución de la población no es la adecuada y tras 5000 generaciones sólo llega a alcanzar valores máximos del 45 %. A pesar de ello parece que el porcentaje podría continuar creciendo si se dejase ejecutar durante un número mayor de iteraciones.

- Mutación 0,02

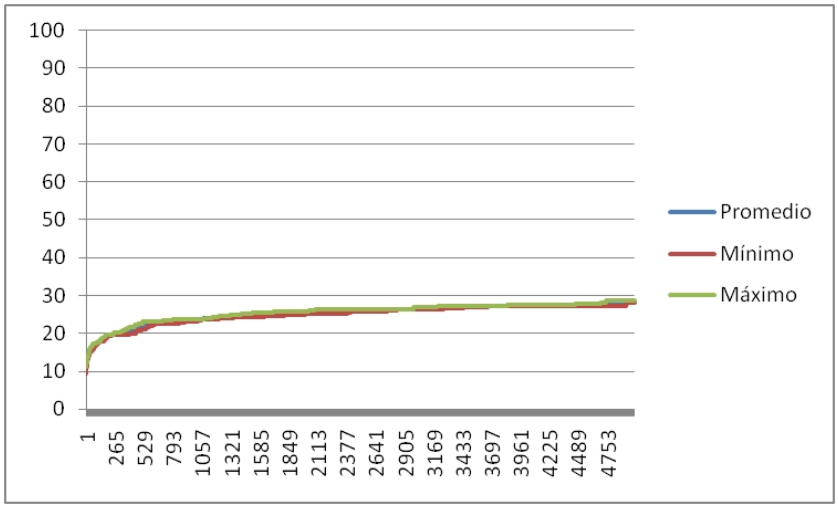

Figura 6.11: Evolución de la población con mutación clásica de 0,02

Analizando la figura 6.11 se confirma el hecho atisbado en los primeros experimentos en los que cuanto mayor es el porcentaje de cruce, peores son los resultados obtenidos. En este caso esos resultados rondan el 27 %, claramente insuficientes aunque no se comparase con cualquier otro posible valor de fitness obtenido por el sistema.

- Mutación 0

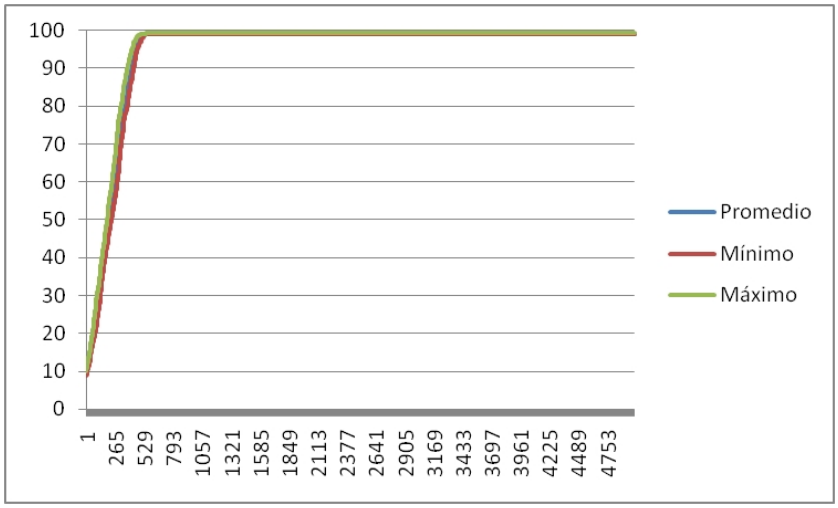

Figura 6.12: Evolución de la población sin utilizar mutación

Se comprueba como nuevamente, la evolución de la población sin el empleo de la mutación es muy rápida y precisa pese a haber incrementado notablemente el número de notas, alcanzando valores muy próximos al máximo en aproximadamente 600 generaciones. La comparación con los otros porcentajes de mutación es indudablemente muy favorable a éste y se postula como favorita para las pruebas posteriores.

Para terminar las comprobaciones del porcentaje de mutación idóneo que se debería utilizar, se va a variar de forma drástica la configuración impuesta hasta ahora con el fin de cerciorarse de las razones que hacen que el sistema evolucione mejor sin mutación. Por ello, se planteará al sistema la búsqueda de una cadena generada aleatoriamente formada por 3 pistas de 1000 notas cada una. También se modificará la función de fitness otorgando un punto positivo por cada bit que se encuentre en la posición correcta y cero para aquella que no. Se dejarán fijos tanto el cruce por compás como la selección por torneo de 5 individuos. Con esta prueba comprobaremos si los resultados obtenidos anteriormente son consistentes o si por el contrario está produciéndose algún error. Además, en caso de ser consistentes, ayudará a proporcionar una explicación razonada de la situación.

A continuación se muestran las evoluciones de las poblaciones bajo la configuración explicada en el párrafo previo:

- Mutación 0

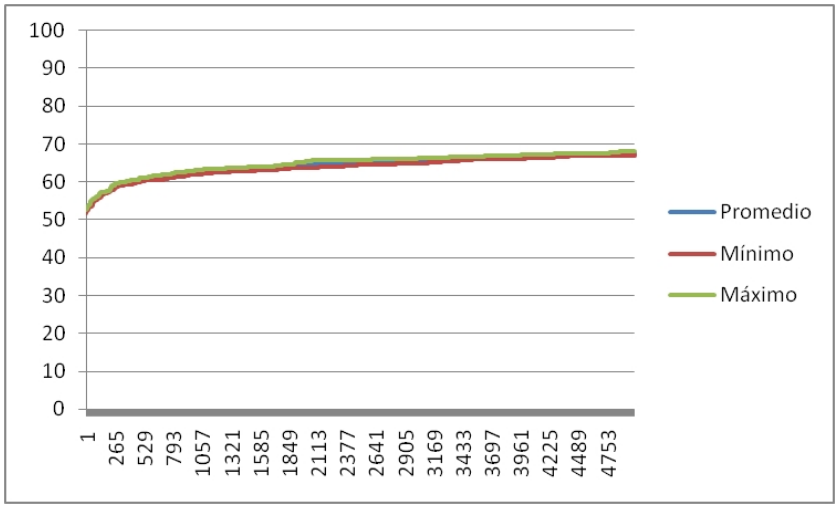

Figura 6.13: Evolución de la población con mutación 0,0

Se comprueba como en este caso la evolución es positiva pero con una tendencia mucho más suave. Alcanza valores cercanos al 70 % tras 5000 generaciones, pese a que parece que aún se encontraba en evolución, lo que hace pensar que los resultados mejorarían aumentando el tiempo de ejecución del sistema.

- Mutación 0,01

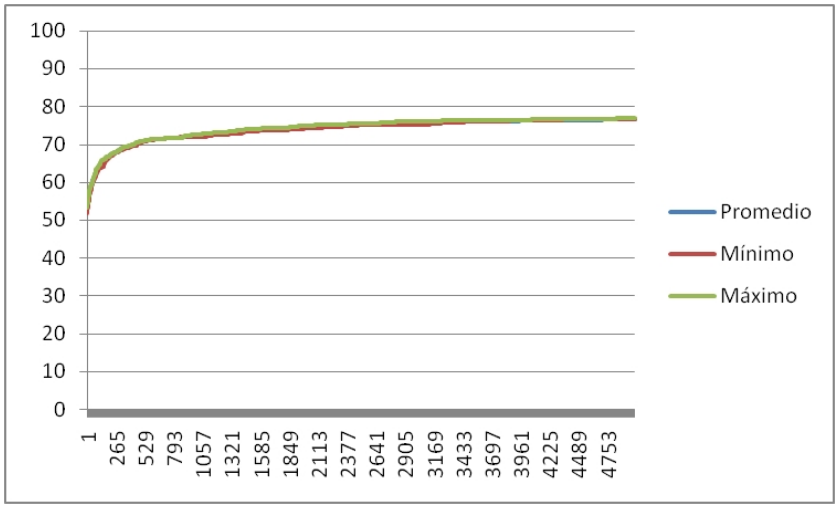

Figura 6.14: Evolución de la población con mutación 0,01

La curva descrita por la gráfica 6.14 muestra la clásica evolución de estos sistemas, más rápida en un principio para seguir posteriormente evolucionando de forma más estable. Los resultados aquí si que concuerdan con lo esperado. Con un valor de mutación aceptable, el sistema evoluciona de una manera más favorable que careciendo de ella.

- Mutación 0,02

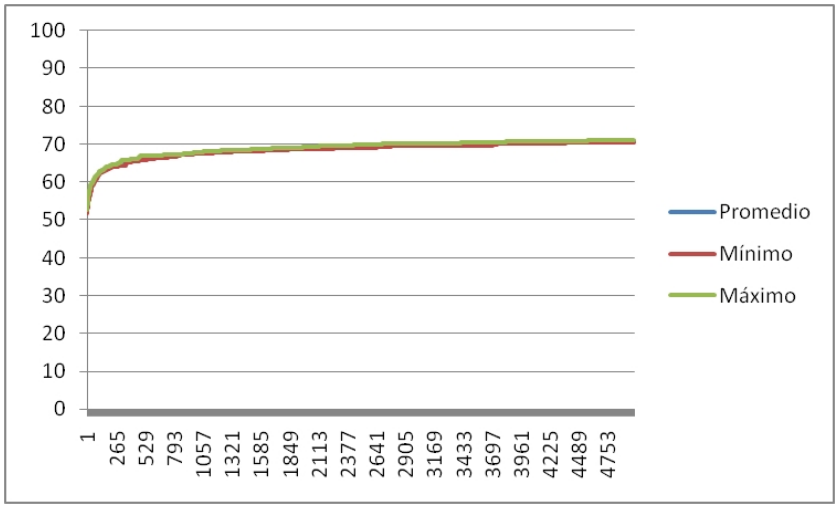

Figura 6.15: Evolución de la población con mutación 0,02

En la figura 6.15 se observa como la línea que sigue el promedio de las poblaciones estudiadas es muy similar al del caso anterior, lo que sigue lo esperado y deseado, pero obteniendo peores resultados, lo que indica que, en este caso, un porcentaje de mutación más elevado no tiene por qué ser más favorable.

Los resultados obtenidos de estas pruebas permite aclarar lo sucedido en la situación anterior. En aquel caso, el tipo de fitness que se está calculando hace que el cruce por compás funcione extremadamente bien, tan bien que el hecho de que la mutación varíe lo que ha conseguido el cruce, perjudica al sistema de forma considerable. En otras palabras, al estar ante un sistema que valora la repetición de patrones y a la vez tener un cruce que trabaja y corta esos mismos patrones, hace que cuando se encuentra un patrón "bueno" y que además se repite, se produzca una propagación rapidísima del mismo por los individuos que forman la población.

De ahí se deriva también el resultado obtenido durante la elección del cruce, ya que como muestra la siguiente figura, en el caso de la búsqueda de la cadena aleatoria el cruce multipunto con una mutación de 0,01 % funciona en algunos casos incluso mejor que el cruce por compás (véase figura 6.16).

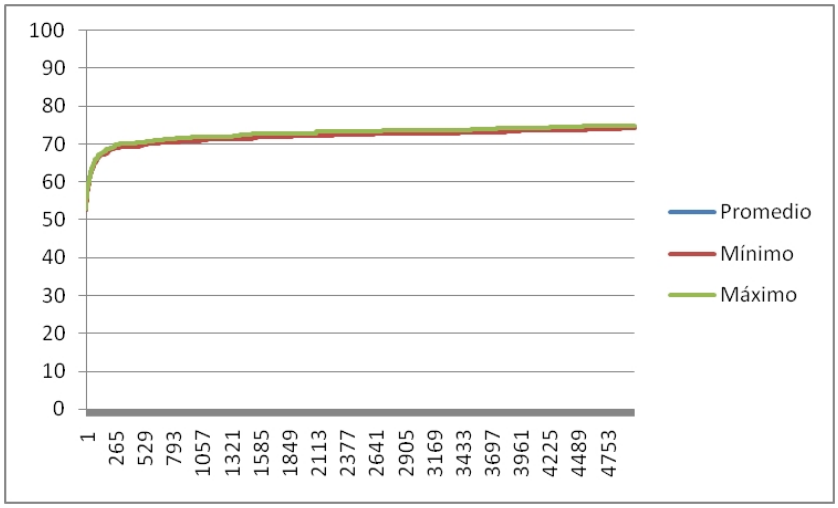

Figura 6.16: Búsqueda de cadena aleatoria de 1000 notas mediante cruce multipunto y mutación de 0,01

A continuación se adjunta la tabla resumen 6.3 con los parámetros manejados en las pruebas anteriores:

Mutación	Fitness	Cruce	Nº notas	Promedio	Mínimo	Máximo
0,0	Patrones	Compás	1000	99,1336	99,02546	99,34987
0,01	Patrones	Compás	1000	42,61071	41,34204	44,56931
0,02	Patrones	Compás	1000	28,25777	28,03846	28,68729
0,0	Cadena	Compás	1000	67,5	67,23333	68
0,01	Cadena	Compás	1000	76,76667	76,56667	77,03333
0,02	Cadena	Compás	1000	70,72222	70,5	70,93333
0,01	Cadena	Multipunto	1000	74,46667	74,2	74,8

Cuadro 6.3: Resumen configuración mutaciones y resultados

La tabla 6.3 muestra muy claramente la información explicada anteriormente. Se puede comprobar como cuando el sistema utiliza el fitness por patrones, siempre combinado con el cruce por compás, se obtienen los mejores resultados (al no existir mutación), mientras que cambiando el tipo de fitness se observa que la no existencia de mutación empeora los resultados alcanzados. Además, cambiando el tipo de cruce empleado sobre el fitness de la cadena aleatoria, se observa como los resultados son muy similares a los obtenidos con el mismo porcentaje de mutación y el cruce por compás, y además son superiores a aquellos también con cruce por compás y sin mutación.

Toda esta información nos lleva a la conclusión de que seleccionando el cruce por compás se debe de utilizar una mutación de 0 para obtener los mejores resultados. La mayoría de las pruebas por tanto seguirán esta configuración, si bien en algunos momentos se pueden incluir pequeños porcentajes de mutación para comprobar la forma en la que afectan a los resultados y obtener algunas conclusiones adicionales.

6.2. Sistema de creación automático

En esta sección se procederá a mostrar y explicar las pruebas más interesantes o relevantes que se han realizado en el marco de la versión automática del sistema así como a extraer las conclusiones precisas en cada caso.

Para poder observar y evaluar la evolución del sistema y los cambios sufridos a lo largo de todo el proceso, los experimentos han sido divididos según la función de fitness que se ha aplicado en cada caso. De esta manera se hará más sencillo tanto el estudio de cada una de ellas por separado como conjuntamente, pudiendo compararlas sin necesidad de modificar la información extraída.

Para la gran mayoría de las pruebas han sido utilizados los operadores seleccionados como óptimos en los apartados anteriores, si bien en algunos experimentos concretos han podido ser modificados por intereses concretos en la investigación. En cualquier caso se indicarán los cambios en la propia configuración cuando sea necesario. Lo mismo ocurre para el valor que se ha otorgado a cada tipo de fitness. Se han estudiado muy diversas combinaciones de los mismos y los efectos de estos cambios en las bases rítmicas resultantes y aquí se mostrarán aquellos que se han considerado más adecuados y provechosos para lograr los objetivos del proyecto.

Las pruebas que se ejecuten serán las mismas para todos los casos, para poder comparar fácilmente unos resultados con otros y observar las dife-

rencias que se producen en los individuos generados, comprobando que se consiguen los objetivos marcados en cada caso. Todas ellas constarán de 30 ejecuciones por configuración planteada. También se mantendrán fijos, salvo indicación expresa, la utilización de la selección por torneo de 5 individuos y el cruce por compás.

Un aspecto más que se mantendrá a lo largo de todas las pruebas es la inclusión de una función que comprueba que ninguna de las pistas que componen una canción esté compuesta íntegramente por ceros, es decir, que exista una o varias pistas en la canción que no reproduzcan ningún sonido. Si esto ocurre, el sistema otorgará a esa pista un fitness de 0, con lo que las posibilidades de que sea seleccionada son muy pequeñas y el individuo en cuestión se perderá en las siguientes generaciones.

Una vez comentada toda la información referente al entorno en el que se realizarán las pruebas, se presentan las diferentes configuraciones que se probarán para cada tipo de fitness:

1. Prueba 1
    - 3 pistas
    - 100 notas por pista
    - Compás 2/4
    - Mutación 0,0
2. Prueba 2
    - 3 pistas
    - 100 notas por pista
    - Compás 2/4
    - Mutación 0,01
3. Prueba 3
    - 3 pistas
    - 100 notas por pista
    - Compás 4/4
    - Mutación 0,0
4. Prueba 4
    - 3 pistas

- 1000 notas por pista
- Compás 2/4
- Mutación 0,0

Además de las pruebas o configuraciones mencionadas, en cada caso particular se pueden presentar pruebas adicionales que proporcionen resultados interesantes o merecedores de mención.

Como información adicional, remarcar que los valores de todas las pruebas se encuentran ponderados, de forma que, sea cual sea el valor total del fitness alcanzado por los individuos, se representará como un porcentaje de entre 0 y 100.

Con el objetivo de simplificar este documento y evitar la repetición de información y la monotonía, se han omitido los resultados obtenidos para la primera función de fitness desarrollada que únicamente tenía en cuenta los patrones principales. Se ha tomado esta decisión debido a que los resultados obtenidos son muy similares a los de la función que tiene en cuenta tanto los patrones principales como los secundarios y no aportan datos relevantes.

Sin más dilación se procede a detallar los conjuntos de pruebas mencionados.

6.2.1. Fitness basado en patrones principales y patrones secundarios

A continuación se presentan las pruebas correspondientes a la ejecución del sistema empleando las funciones de fitness que valoran los patrones principales y secundarios que encuentra a lo largo de una pista.

Los siguientes resultados que se presentan se corresponden a los obtenidos empleando una puntuación de 100 por cada patrón principal encontrado y de 1 por cada patrón secundario. Con estos valores del fitness claramente se está dando mucha prioridad al hecho de encontrar patrones principales sobre los secundarios. Estos valores representan, por tanto, una forma lógica de representación, ya que es más adecuado buscar resultados que se ajusten al compás en el que se mueve la canción y, aparte, dar una puntuación adicional si además se adaptan a alguno de los ritmos compatibles. De otra manera premiando al individuo de forma más pareja se podría obtener como resultado un patrón que fuese más adecuado para otro tipo de compás.

Prueba 1 (P=3, N=100, C=2/4, M=0)

En esta ocasión se comprobará la forma en que el sistema responde a este valor de las funciones del fitness con mutación nula, en un compás 2/4

y con una longitud de 100 notas por pista.

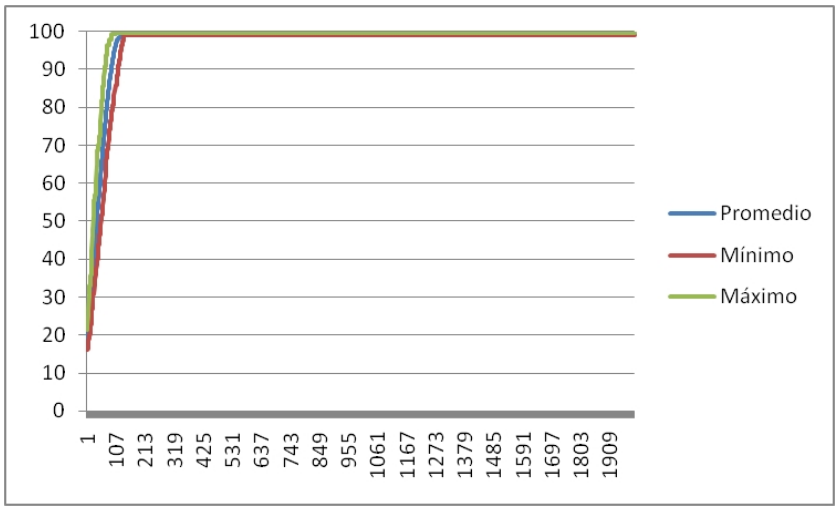

Figura 6.17: Evolución de la población con fitness principal y secundarios en prueba 1

En la figura 6.17 se observa con claridad como el sistema responde de forma casi inmediata y alcanza resultados muy positivos cerca de la generación 200. La rápida convergencia de la prueba tiene su explicación en lo comentado en la sección anterior. Al ser el cruce por compás un elemento que se adapta perfectamente al sistema, los valores adecuados se propagan de una forma muy rápida y provocan la aparición de individuos casi perfectos desde el punto de vista del fitness.

En las figuras 6.18 y 6.19 se puede observar un par de ejemplos de los individuos generados:

Pista 1	1	0	0	1	1	0	0	1	1	0	0	1
Pista 2	0	1	0	1	0	1	0	1	0	1	0	1
Pista 3	0	0	1	0	0	0	1	0	0	0	1	0

Figura 6.18: Ejemplo de individuo resultante de prueba 1

En el ejemplo de la figura 6.18 se muestran los 12 primeros bits de los 100

que componen la pista. Es posible observar como al tratarse de un compás 2/4 los bits se han agrupado en grupos de 4 y repiten continuamente un mismo patrón. Se ve también que al tener en cuenta los patrones secundarios y ser el 1/2 un patrón compatible con el 2/4, si se cogen los bits de dos en dos la segunda pista tendría un valor más alto que la primera y la tercera en la que no ocurre así. El 4/8 se considera también un compás compatible y si se cogen grupos de 8 vemos como todas las pistas lo cumplen al repetirse siempre los patrones de longitud 4.

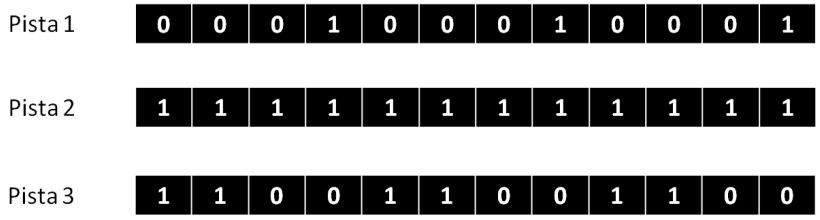

Figura 6.19: Ejemplo de individuo resultante con prueba 1

El caso de la figura 6.19 es exactamente igual al anterior. Se repite siempre el mismo patrón en cada grupo de 4 y 8, y en la segunda pista además cada 2 al ser una pista formada íntegramente por unos.

Los individuos mostrados reflejan lo que se ha querido conseguir empleando esta función de fitness, sin embargo ninguno de estos dos ejemplos escogidos poseería un fitness máximo. Si lo poseería un individuo con pistas que repitiesen un patrón de 2 bits de longitud (que evidentemente se han obtenido en las pruebas realizadas) aunque esto haría que no se diferenciasen de compases x/2 y el hecho de puntuar de forma más elevada los patrones principales hace reducir las posibilidades de que se produzca siempre ese caso, aumentando así, la variedad de soluciones que se pueden conseguir.

Prueba 2 (P=3, N=100, C=2/4, M=0,01)

En este caso se decide repetir la prueba anterior pero esta vez incluyendo un porcentaje de mutación de 0,01. En las pruebas de los operadores genéticos se ha comprobado como la mutación no sólo no afecta positivamente sino que en muchos casos se torna perjudicial para la evolución de la población.

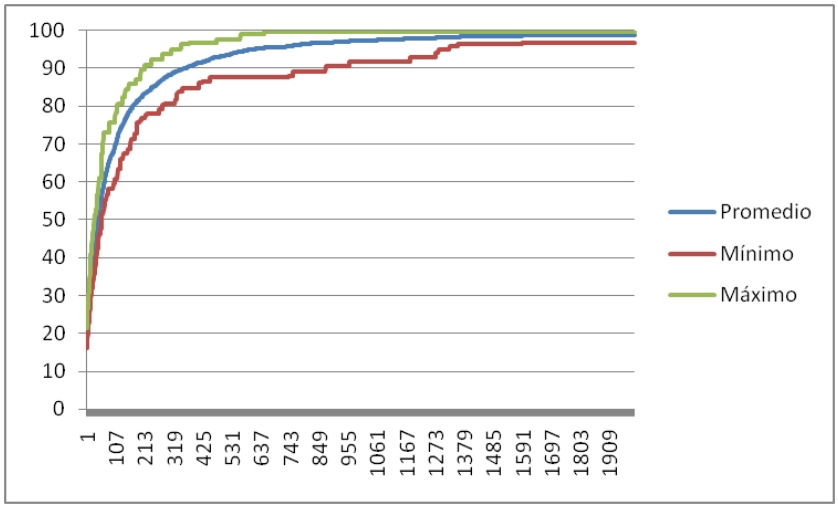

Figura 6.20: Evolución de la población con fitness principal y secundarios en prueba 2

Comparando los resultados obtenidos con el caso anterior, es fácil observar en la figura 6.20 como la evolución de la población es algo más lenta, no obteniendo un fitness promedio similar al conseguido en la prueba anterior hasta la generación 1500 aproximadamente. Se constata una vez más el perjuicio infligido por la mutación y que indica que en longitudes de pista mayores pueda impedir el desarrollo completo de los individuos de la población de la manera deseada.

Los mejores individuos obtenidos siguen la línea de los mostrados en el caso anterior, ya que aunque se tarda más en alcanzarlos, el valor final de los dichos individuos es el mismo.

Prueba 3 (P=3, N=100, C=4/4, M=0)

Para este caso se ha decidido volver a una mutación nula pero se ha cambiado el compás que pasa a ser 4/4 en lugar del 2/4 utilizado en las pruebas anteriores.

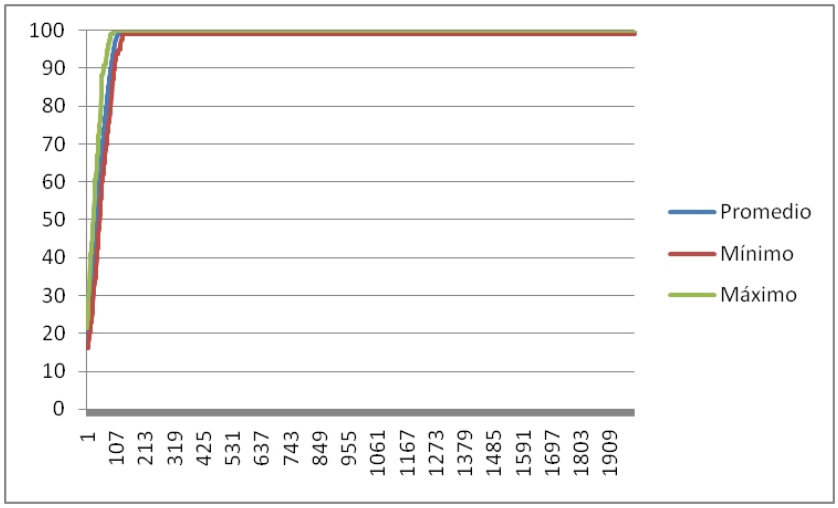

Figura 6.21: Evolución de la población con fitness principal y secundarios en prueba 3

Se comprueba en la figura 6.21 como la evolución de la población es muy similar a la del caso del compás 2/4 muy rápida, convergiendo la función alrededor de la generación 200.

La figura 6.22 representa un ejemplo cogido al azar de uno de los mejores individuos obtenidos en las diversas ejecuciones realizadas con esta configuración:

Pista 1	1	1	0	0	1	1	0	0	1	1	0	0
Pista 2	0	1	1	1	0	1	1	1	0	1	1	1
Pista 3	0	1	1	0	0	1	1	0	0	1	1	0

Figura 6.22: Ejemplo de individuo resultante con prueba 3

El resultado obtenido es igual al caso del compás 2/4. Este es un resultado esperado, ya que esta función no tiene en cuenta en ningún momento el primer número que forma el compás y sólo intenta agrupar los patrones de la longitud que marque el segundo número del compás y los de sus patrones compatibles. La diferencia entre estas dos pruebas se hará patente al aplicar la función de fitness que se tratará en la próxima sección.

Prueba 4 (P=3, N=1000, C=2/4, M=0)

Con esta prueba se pretende comprobar el rendimiento del sistema aumentando considerablemente el número de notas que formará cada pista (1000), comprobando si esta cantidad puede influir en los resultados que se obtienen. Para ello se recupera el compás 2/4 y se continúa con mutación 0.

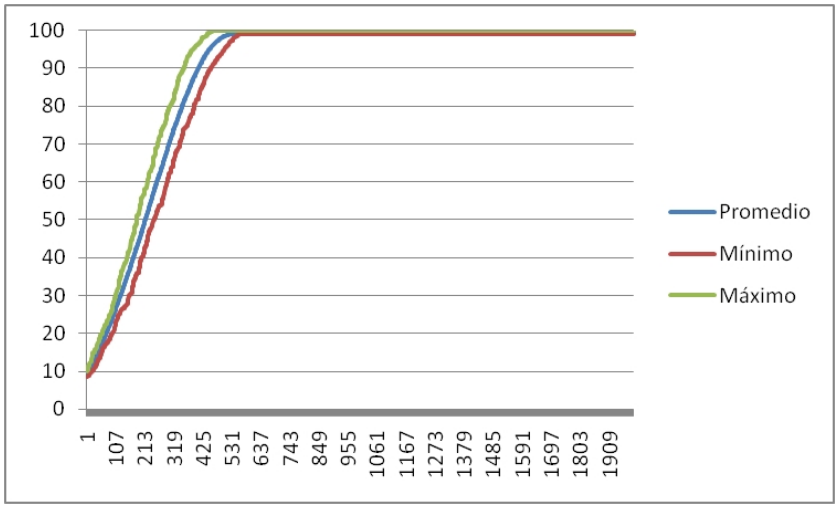

Figura 6.23: Evolución de la población con fitness principal y secundarios en prueba 4

Se observa en la figura 6.23 como el hecho de aumentar hasta un número muy elevado la longitud de notas de la pista no afecta al buen rendimiento del sistema. Se alcanzan valores muy próximos al máximo en generaciones tempranas (aunque lógicamente mayores que en la prueba 1).

Los individuos obtenidos continúan siendo similares, solo que esta vez en lugar de repetir un patrón hasta completar las 100 notas, lo hacen hasta completar las 1000.

Prueba 5 (P=3, N=1000, C=2/4, M=0,01)

Para confirmar la afirmación que se realizó con anterioridad del hecho de que la mutación puede comprometer el éxito de la función con espacios de búsqueda mayores se ha decidido incluir un experimento más en esta batería de pruebas consistente en utilizar una longitud de notas de 1000 y una mutación de 0,01 manteniendo el compás 2/4 y el mismo número

de generaciones. Tras realizar esta prueba se podrá comprobar el grado de influencia de la mutación en el sistema.

La figura 6.24 muestra los resultados del experimento:

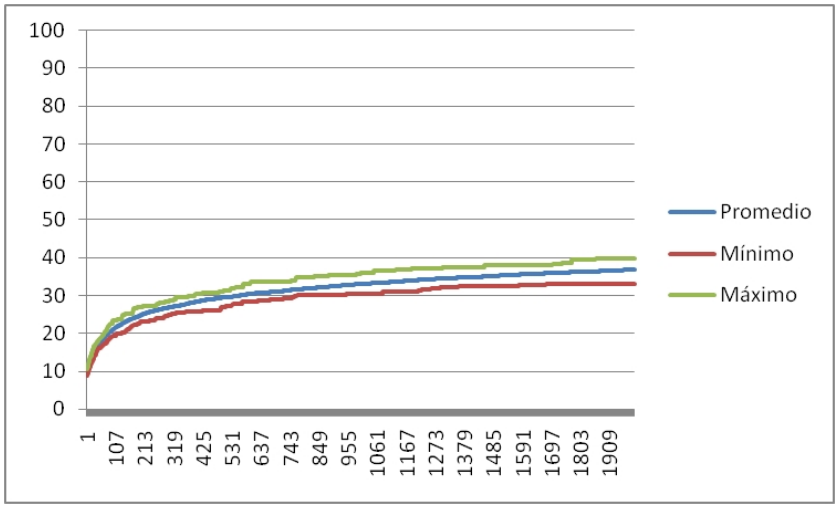

Figura 6.24: Evolución de la población con fitness principal y secundarios en prueba 5

Los resultados precisan pocos comentarios. En este caso y tras completar las 2000 generaciones propuestas, el máximo que alcanza la función es de cerca de un 40 %, valor muy por debajo de lo esperado y que refleja el gran efecto negativo provocado por la mutación pese a ser de un valor muy pequeño. La explicación, como se comentaba anteriormente y se ha detallado en la sección de operadores genéticos de este capítulo, se corresponde con la acción conjunta de una función de fitness que evalúa por compases y un cruce que también actúa directamente sobre esos compases.

Conclusiones

Los resultados que han sido obtenidos tras la realización de las pruebas son en general muy buenos. Los individuos resultantes cumplen los objetivos marcados por la función de fitness empleada, generando bases rítmicas con un orden claro y adaptado al compás elegido por el usuario.

Otro punto importante es la diversidad de las soluciones generadas, ya que el abanico de patrones que cumplen los requisitos de la función de fitness es muy amplio y además se deben tener en cuenta las posibles combinaciones

de cada uno de esos patrones con el resto de las pistas que formarán la canción.

La siguiente tabla resume los resultados obtenidos en las pruebas de esta función de fitness:

	Mutación	Compás	Nº notas	Promedio	Mínimo	Máximo
Prueba 1	0,0	2/4	100	99,28194	99,00078	99,64962
Prueba 2	0,01	2/4	100	98,88874	96,69089	99,64962
Prueba 3	0,0	4/4	100	99,21728	99,00078	99,97405
Prueba 4	0,0	2/4	1000	99,31743	99,02546	99,9987
Prueba 5	0,01	2/4	1000	36,72268	32,93452	39,76409

Cuadro 6.4: Resultados pruebas de fitness con patrones principales y secundarios

La tabla 6.4 muestra claramente la gran adaptación de los individuos al sistema. Únicamente se sale de esta línea empleando la mutación en longitudes de notas muy grandes por el hecho ya comentado de que la mutación "deshace" lo conseguido por el cruce.

Un factor importante es que la gran variedad en las soluciones y la diversidad de los individuos conseguidos pueden estar íntimamente relacionados con la existencia de pocas restricciones impuestas por la función de fitness. Este interrogante será resuelto a medida que se avanza en las pruebas, ya que en cada conjunto de ellas se irá incluyendo una nueva función que añadirá complejidad al sistema y por tanto modificará los resultados obtenidos en los tests anteriores.

6.2.2. Fitness basado en patrones y ritmos

En este punto se comentarán las pruebas realizadas empleando la función de fitness de patrones principales y secundarios incluyendo también la función de fitness que añade el carácter rítmico a la canción generada al intentar asegurar que se produzca un sonido en el lugar donde se debe marcar el ritmo, que como bien es sabido, se produce al inicio de cualquier compás.

Es por tanto en este punto donde empieza a tenerse en cuenta el primer número del compás.

Los resultados presentados han sido obtenidos empleando una puntuación de 100 por cada patrón principal encontrado, de 1 por cada patrón secundario y de 50 por cada vez que el golpe del ritmo se produzca en el lugar adecuado. Con estos valores para cada parte del fitness se continúa la línea del apartado anterior y además se añade un peso importante a que el ritmo quede marcado correctamente. Esto es así debido a que cualquier pieza rítmica precisa de seguir el orden marcado en un porcentaje elevado del tiempo en el que se reproduzca, aunque a veces puedan existir variaciones. Además de esta manera se consiguen distinguir compases como el 2/4 y el 4/4, hecho que en los casos anteriores no era posible y que se considera básico y necesario.

Sin más preámbulos se detallan los resultados obtenidos tras realizar las pruebas ya comentadas:

Prueba 1 (P=3, N=100, C=2/4, M=0)

Nuevamente se comprobará la forma en que el sistema responde a este valor de las funciones del fitness con mutación nula, en un compás 2/4 y con una longitud de 100 notas por pista.

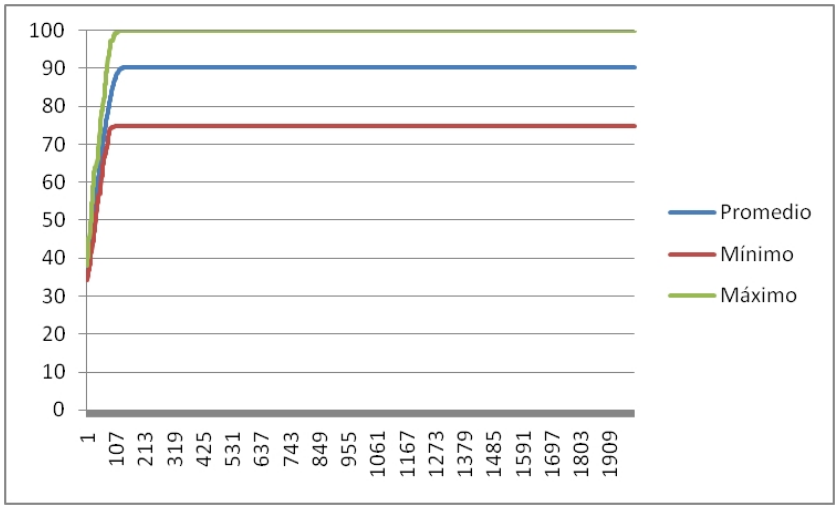

Figura 6.25: Evolución población con fitness principal, secundarios y ritmo en prueba 1

El comportamiento de la población continúa siendo positivo, como se observa en la figura 6.25, pero en este caso no se alcanza el máximo en todos los individuos generados. Aunque la evolución es muy rápida, ésta se para en un determinado punto y permanece ahí. Este hecho es debido a la inexistencia de mutación, que en algunos casos hace que la función se estanque en un mínimo local y no pueda salir.

Debido a la naturaleza del problema, este hecho puede considerarse en algunos casos como positivo, ya que variaciones en el orden prestablecido pueden resultar de gran interés en el mundo de la música, como pueden ser las variaciones o improvisaciones.

En las figuras 6.26 y 6.27 se presentan dos ejemplos de individuos generados con esta configuración:

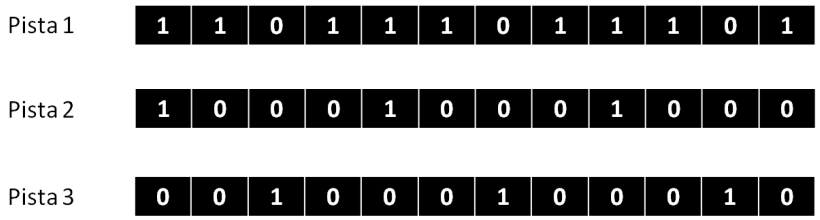

Figura 6.26: Ejemplo 1, individuo obtenido en prueba 1

La figura 6.26 muestra uno de los peores individuos alcanzados tras la evolución de la población. Es fácil observar como sigue siempre la línea de los casos anteriores, repitiendo un mismo patrón principal a lo largo de toda la pista, pero en este caso no se cumple siempre la característica que marca la nueva función de fitness incorporada. Al estar en un compás 2/4, se debería marcar el ritmo en las posiciones 0, 2, 4, 6... y se observa como no es lo que ocurre en este caso. El siguiente ejemplo difiere algo del presentado aquí:

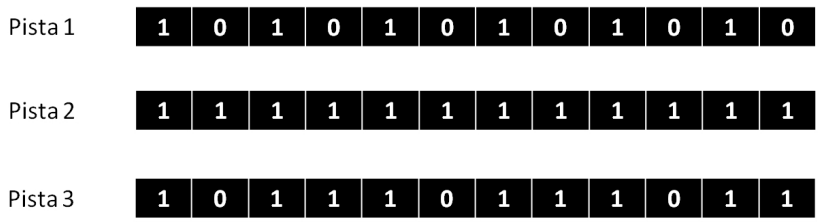

Figura 6.27: Ejemplo 2, individuo obtenido en prueba 1

Este segundo ejemplo, figura 6.27, representa uno de los mejores indi-

viduos obtenidos en las pruebas con esta configuración. La diferencia con el anterior es que, además de repetir los patrones principales a lo largo de toda la longitud, la nueva función de fitness ha valorado positivamente el hecho de que en todas las posiciones que se debe marcar el ritmo, se hace, ya que en las tres pistas que componen el individuo aparece un "1" en las posiciones clave comentadas anteriormente. Aún así, este no sería el mejor individuo que se podría obtener con estas funciones de fitness ya que, a pesar de que la primera y la segunda pista cumplen también los requisitos de los patrones secundarios, no lo hace la tercera, que no concuerda con el compás compatible x/2.

Como se ha comentado a lo largo de todo el documento, estos hechos no siempre tienen que ser considerados como negativos a pesar de que impidan alcanzar el fitness máximo, ya que la variación también puede ser positiva en el ámbito musical, creando composiciones más originales y no tan estrictamente ceñidas en su totalidad a las reglas marcadas.

Prueba 2 (P=3, N=100, C=2/4, M=0,01)

Se vuelve a utilizar la configuración anterior pero cambiando la mutación, que toma un valor cero para, de esta forma, seguir comprobando el modo en que este valor afecta al conjunto de los resultados obtenidos. El siguiente gráfico se encarga de mostrarlos:

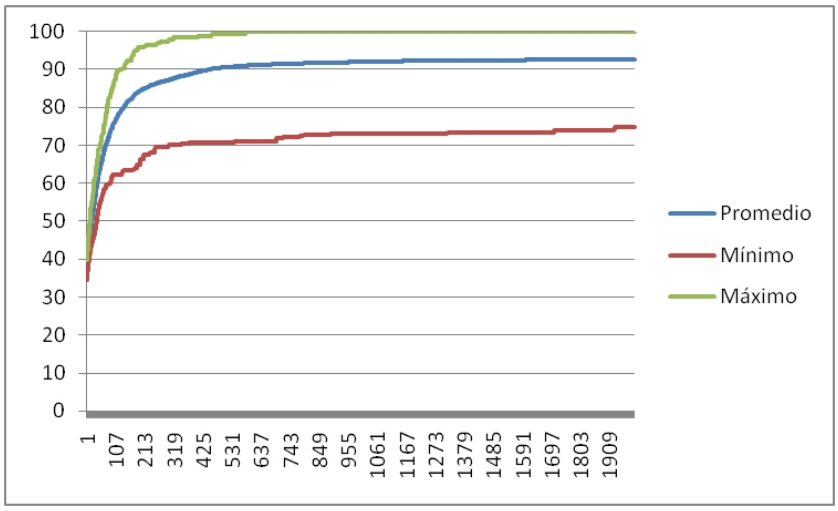

Figura 6.28: Evolución población con fitness principal, secundarios y ritmo en prueba 2

Si se compara la figura 6.28 con la que se comentó en la prueba anterior (figura 6.25), se puede observar como siguen una línea muy similar. La evolución es más lenta pero, en promedio, llega a alcanzar un valor más alto al que se obtuvo sin mutación. Este hecho no se había producido en ninguno de los experimentos anteriores, donde las ejecuciones sin mutación eran claramente superiores. Este cambio se debe a que al ir aumentando los factores que se tienen en cuenta a la hora de evaluar los individuos, empieza a perder algo de peso el compás obtenido en un principio en favor del resto de parámetros evaluables. De esta forma puede resultar que la mutación comience a ser más necesaria a partir de ahora, influyendo menos en la "destrucción" de los patrones creados por el cruce y evitando el estancamiento en máximos locales que afectan a las poblaciones carentes de mutación. Para comprobar esta situación se estudiarán el resto de pruebas de esta sección y de las siguientes con el fin de establecer unas conclusiones más precisas y basadas en la experiencia.

Prueba 3 (P=3, N=100, C=4/4, M=0)

La siguiente prueba es una de las estándar comentadas anteriormente y consiste en cambiar el compás a un 4/4 y volver a mutación nula para comprobar como se adapta el sistema a diversas configuraciones:

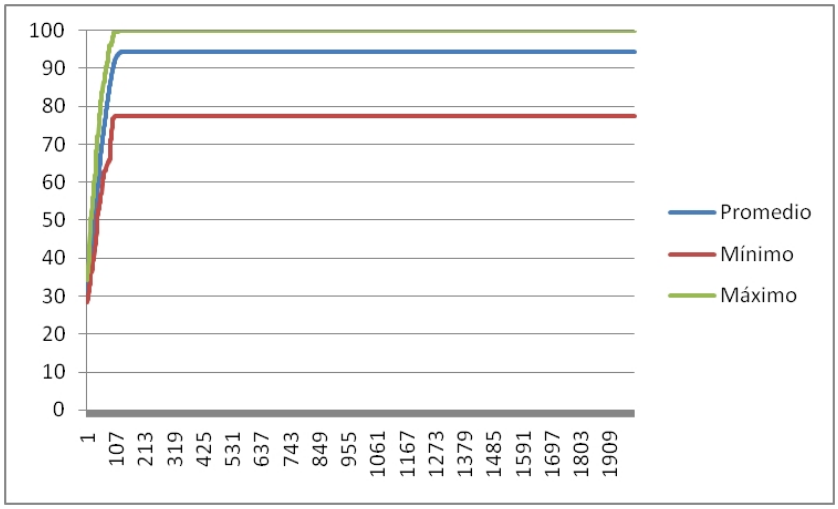

Figura 6.29: Evolución población con fitness principal, secundarios y ritmo en prueba 3

Se observa en la figura 6.29 como los resultados continúan de la misma manera que antes. La evolución con mutación nula es muy rápida y además consigue resultados muy buenos que, en este caso, superan a los conseguidos en la primera prueba empleando un compás 2/4. ¿Por qué actúa mejor con este tipo de compás?

La solución se debe a dos motivos. El primero de ellos es que los patrones que se están buscando poseen longitud 4, al igual que lo que ahora es el valor del primer número del compás. Esto hace que sea algo más sencillo para el sistema encontrar patrones que además cumplan el requisito de que el golpe del ritmo se produzca al inicio del compás. La segunda razón, y la más poderosa, responde a que al ser el primer número del compás mayor en este caso que en la prueba 1, hay menos posiciones que comprobar y, por tanto, menos posiciones del array de notas que deben cumplir este requisito, resultando más fácil satisfacer la condición alcanzando valores de fitness más altos.

En pruebas sucesivas se intentarán comprobar estos datos con experimentos adicionales y complementarios.

Prueba 4 (P=3, N=1000, C=2/4, M=0)

Para esta prueba seguimos con la batería de pruebas establecida y se vuelve a aumentar considerablemente el número de notas que posee cada pista (hasta 1000) para así comprobar como responde el sistema ante esta modificación.

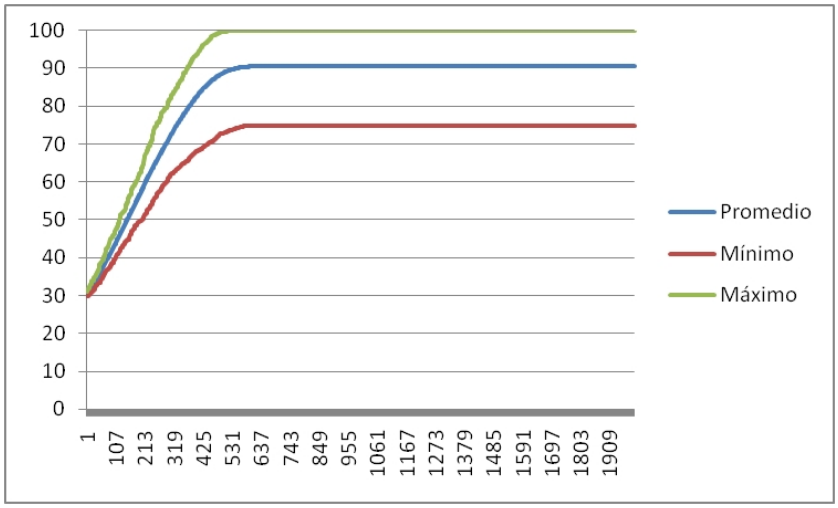

Figura 6.30: Evolución población con fitness principal, secundarios y ritmo en prueba 4

Como muestra la figura 6.30, los resultados siguen siendo estables y siguen sin apreciarse cambios significativos en la evolución de la población aumentando el número de notas por pista, siempre manteniendo la mutación en 0. El valor promedio alcanzado es prácticamente igual al conseguido con la misma configuración y 100 pistas por nota, lo que demuestra esa independencia del número de notas a la hora de obtener buenos resultados.

Prueba 5 (P=3, N=1000, C=2/4, M=0,01)

Para estudiar más a fondo las consecuencias de la introducción de valores pequeños de mutación, se ha decidido repetir la prueba 5 realizada también para la función de fitness anterior. Esta configuración se corresponde con 3 pistas, 1000 notas por pista y mutación de 0,01 %. Con ello se podrá comprobar si el hecho de que haya funcionado mejor con ese valor de mutación que con 0 para una longitud de 100 notas se puede hacer extensible a todos los casos en esta función de fitness.

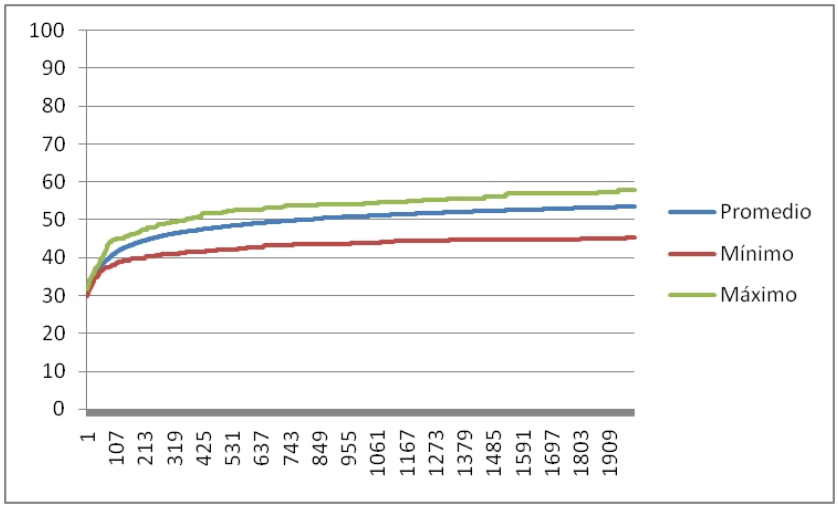

Figura 6.31: Evolución población con fitness principal, secundarios y ritmo en prueba 5

Se comprueba observando la figura 6.31 como al aumentar el número de notas, el uso de la mutación empeora los resultados obtenidos. Aún así, estos son un 20 % mejores que los conseguidos con la función de fitness anterior, lo que no deja duda a que algo sí que ha cambiado, a pesar de que no sean resultados aceptables para los objetivos planteados. Se seguirá estudiando la evolución de esta configuración en las pruebas por venir con el fin de seguir de cerca los cambios que puedan producirse.

Prueba 6 (P=3, N=100, C=2/2, M=0)

Las pruebas anteriores dejaron preguntas en el aire que pretenden obtener respuesta mediante esta prueba adicional. La prueba 3 arrojó resultados mejores desde el punto de vista del fitness que los obtenidos en la prueba 1. La única diferencia en la configuración de ambas es el compás. Entonces se dió una explicación a esa diferencia, pero se propone esta prueba para corroborar lo detallado en ese punto.

La configuración de la prueba consistirá en individuos de 3 pistas cada uno, con 100 notas por pista y un compás de 2/2. Con ello se probará si la diferencia extraída en la prueba anterior se debe a la coincidencia entre la longitud del patrón y el primer número del compás o al hecho de que a mayor valor del primer número del compás, menores son las comprobaciones y más fácil el cumplimiento del requisito del ritmo.

La figura 6.32 explicará lo ocurrido:

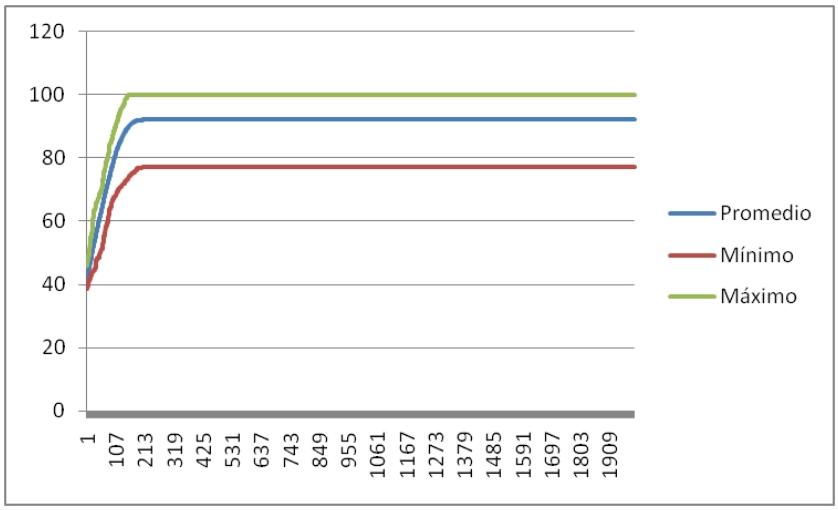

Figura 6.32: Evolución población con fitness principal, secundarios y ritmo en prueba 6

Observando la figura 6.32 se comprueba como los resultados obtenidos se encuentran, en promedio, a medio camino de las situaciones anteriores que se intentan esclarecer. Este hecho hace que no se pueda concluir de forma precisa en la razón exacta de por qué el sistema responde mejor ante un compás 4/4 que ante un 2/4, pero sí se intuye que los motivos que llevan a esa situación se deben más a motivos de sencillez a la hora de alcanzar valores máximos que a combinaciones de compases concretos.

Todo ello refleja que los pesos que se lo otorga a cada valor de fitness influyen en mayor medida. De este modo si al patrón primario se le da un valor de 100 cada vez que se repite y un valor de 50 cada vez que se cumple la regla del primer número del compás, se obtendrán valores muy diferentes en función del tipo de compás empleado. En un 2/2 y en un 4/4 es más fácil encontrar patrones que cumplan ambas partes por tener la misma longitud. Sin embargo en un 2/4, pueden entrar en conflicto los pesos de una y otra función de fitness, lo que puede generar, en promedio, peores individuos.

Conclusiones

Los resultados extraídos de todas las pruebas anteriores han resultado ser muy positivos. Los individuos resultantes cumplen en un alto porcentaje los

objetivos perseguidos añadiendo la nueva función de fitness a las anteriores y continúan generando ritmos adecuados y que cumplen con lo esperado.

Un factor que vuelve a ser interesante es que los valores otorgados a cada tipo de fitness permiten muy diferentes combinaciones en los valores de las pistas, lo que favorece la diversidad y la obtención de soluciones diferentes en cada caso. Con ello también queda claro que otorgando puntuaciones diferentes se obtendrían individuos adaptados a cada situación, por tanto, debe ser le usuario el que priorice las características musicales que desee.

La tabla 6.5 resume los resultados obtenidos en las pruebas de esta función de fitness:

	Mutación	Compás	Nº notas	Promedio	Mínimo	Máximo
Prueba 1	0,0	2/4	100	90,32509	74,8323	99,98685
Prueba 2	0,01	2/4	100	92,54571	74,8323	99,82244
Prueba 3	0,0	4/4	100	94,4105	77,50524	99,98254
Prueba 4	0,0	2/4	1000	90,63913	74,84513	99,99934
Prueba 5	0,01	2/4	1000	53,60083	45,26706	57,74881
Prueba 6	0,0	2/2	100	92,40044	77,27895	99,99116

Cuadro 6.5: Resultados pruebas de fitness patrones principales, secundarios y ritmo

Si se compara el porcentaje obtenido en el conjunto de pruebas de las dos funciones de fitness probadas hasta el momento se puede comprobar como en este caso, prestando más atención al promedio, se encuentran algo por debajo de los anteriores. La explicación de este suceso viene dada por el aumento de las restricciones que tiene en cuenta el sistema a la hora de evolucionar la población de individuos. A medida que estas restricciones o variables aumenten en el sistema para aumentar la calidad y complejidad de los ritmos generados, aumentará también la dificultad a la hora de obtener valores del fitness cercanos al 100 %.

Aún así, los individuos resultantes son correctos y se ciñen a los patrones deseados, resultando en piezas con ritmo marcado y con el orden que se

requiere. Con todo, se llegan a alcanzar valores del fitness superiores al 99 % a pesar de que superado un cierto umbral en el valor del fitness, los individuos generados se corresponden con canciones muy válidas y no es necesario tomar valores tan altos.

Las siguientes secciones seguirán añadiendo complejidad y factores evaluables a las canciones generadas y se continuará estudiando el efecto que tiene cada uno de ellos sobre el total de la población.

6.2.3. Fitness vertical

En este apartado aparecen las diversas pruebas realizadas empleando todas las funciones de fitness anteriores e incorporando la función de fitness que comprueba las pistas en vertical. El fitness vertical se encarga de comprobar que no se produzca gran acumulación de sonidos a la vez durante la reproducción de la canción excepto en las posiciones donde se debe marcar el ritmo, controlado por la función de fitness explicada en la sección anterior. El número de sonidos que pueden sonar de manera simultánea en la canción vendrá determinado por el número de pistas.

Los resultados que se reflejarán a continuación han sido obtenidos empleando las mismas puntuaciones para los casos anteriores; 100 para patrones principales, 1 para patrones secundarios y 50 para cada vez que se produzca el golpe del ritmo en el momento adecuado, y además se incorpora una puntuación positiva de 10 por cada posición en la que se cumpla que se reproducen la cantidad de sonidos adecuados y de -10 por cada posición en la que se reproduzcan más de los permitidos.

Con estos valores se da a cada factor un peso específico dentro de la canción y los resultados serán consecuentes con los valores otorgados.

Sin más dilación, se detallan las diferentes pruebas realizadas y los resultados asociados a ellas.

Prueba 1 (P=3, N=100, C=2/4, M=0)

Se ejecuta nuevamente la primera de las pruebas establecidas cuya configuración se compone de 3 pistas de 100 notas cada una, con un compás de 2/4 y una mutación igual a 0.

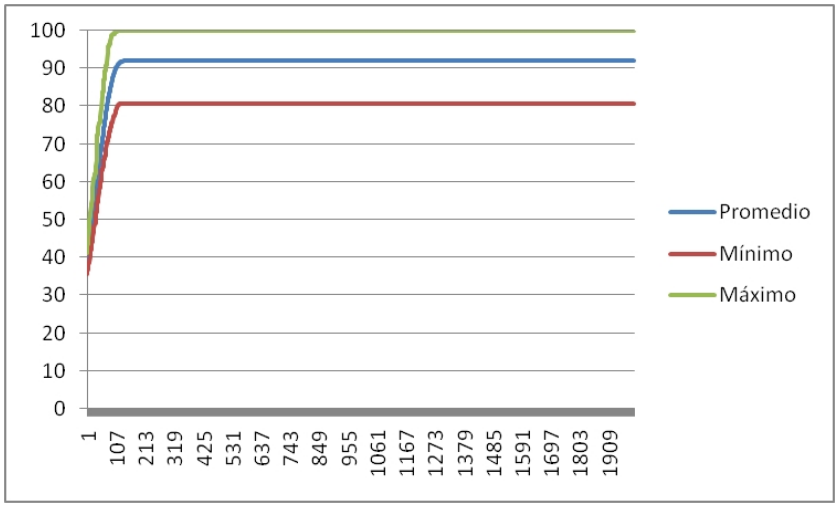

Figura 6.33: Evolución población con fitness principal, secundarios, ritmo y vertical en prueba 1

Se comprueba empleando la figura 6.33 como a pesar de que la complejidad va aumentando a medida que se avanza en las pruebas, el sistema sigue respondiendo de manera adecuada, evolucionando la población de forma positiva y encontrando individuos que satisfacen los criterios deseados.

Las figuras 6.34 y 6.35 muestran dos de los individuos generados tras la evolución del sistema a modo de ejemplo. Así se podrán observar las diferencias existentes con respecto a las funciones de fitness anteriores y el cumplimiento de los requisitos ya comentados:

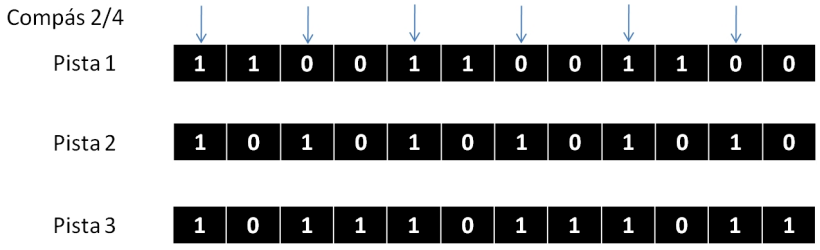

Figura 6.34: Ejemplo 1, individuo obtenido en prueba 1

Las flechas situadas marcan las posiciones en las que se debe marcar el ritmo en un compás 2/4. A simple vista en la figura 6.34 se observa como en las posiciones donde no se marca el ritmo nunca aparecen más de dos

notas sonando a la vez (las correspondientes al valor umbral establecido), mientras que en las posiciones donde existe el golpe de ritmo sí puede darse esta situación, aunque como se ve, no es obligatorio que todas las pistas reproduzcan un sonido en ese instante.

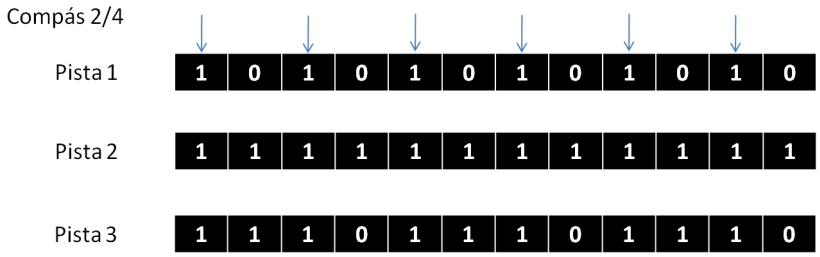

Figura 6.35: Ejemplo 2, individuo obtenido en prueba 1

En este ejemplo representado por la figura 6.35 ocurre lo mismo que en el caso anterior, pero en este sí que en el golpe de compás se produce un sonido siempre en cada pista. En el resto de instantes únicamente suenan una o dos notas de forma simultánea, lo que se encuentra dentro de los permitido por el sistema.

Estos ejemplos ilustran la posibilidad del sistema de seguir añadiendo aspectos musicales a los ritmos sin afectar por ello al rendimiento o a la diversidad de las soluciones generadas. Ambos factores son de vital importancia y permiten que el sistema pueda seguirse desarrollando y aspirando cada vez a mayores cotas.

Prueba 2 (P=3, N=100, C=2/4, M=0,01)

Se realiza esta prueba con la configuración igual a la anterior pero introduciendo un factor de mutación del 0,01 %:

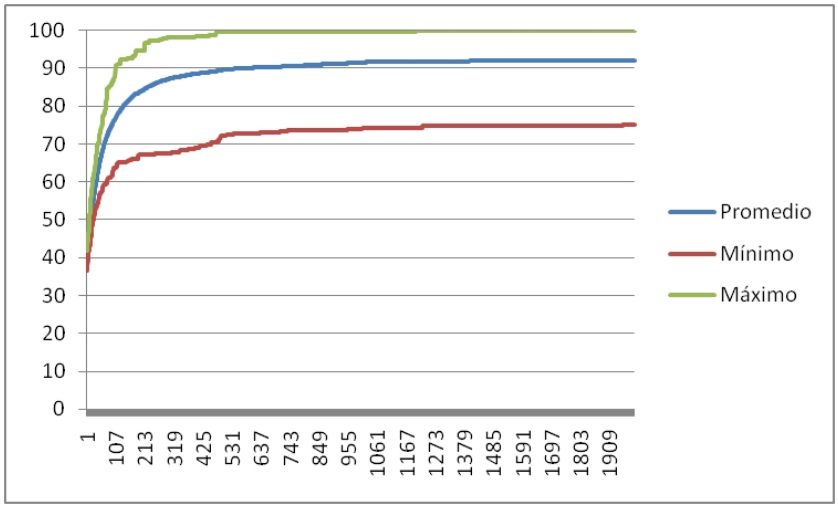

Figura 6.36: Evolución población con fitness principal, secundarios, ritmo y vertical en prueba 2

Como muestra la figura 6.36, los resultados alcanzados son muy similares a los del caso anterior. Se confirma una vez más el hecho de que con valores bajos de número de notas la mutación funciona correctamente mientras que para valores más elevados resulta altamente perjudicial. Otra característica que permanece constante es que el tiempo que se tarda en alcanzar esos valores es claramente superior al precisado por el sistema empleando mutación cero, aspecto a tener en cuenta para futuros desarrollos.

Los individuos generados también permanecen en la misma línea de los mostrados para la prueba anterior.

Prueba 3 (P=3, N=100, C=4/4, M=0)

Se retorna a las pruebas con mutación nula y se altera el tipo de compás, pasando a un 4/4 para observar que el sistema continúa adaptándose de manera correcta a las diversas configuraciones propuestas. La figura 6.37 refleja esta situación:

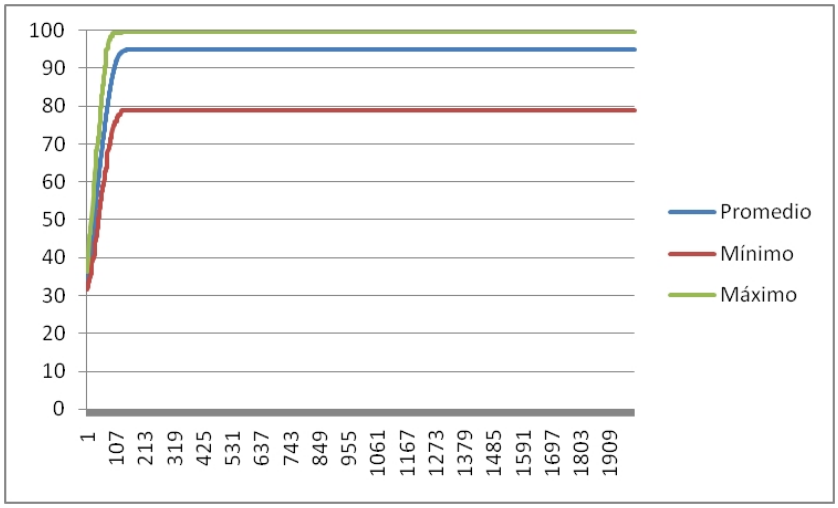

Figura 6.37: Evolución población con fitness principal, secundarios, ritmo y vertical en prueba 3

Como ocurrió en los resultados extraídos de la función de fitness anterior, el valor promedio de los individuos alcanzado por esta configuración está algo por encima del obtenido en la prueba 1. Las razones de esta situación continúan siendo las mismas, ya que la inclusión de esta nueva función de fitness no entra en conflicto con lo comentado para la función de fitness anterior.

La figura 6.38 representa un individuo generado empleando esta configuración:

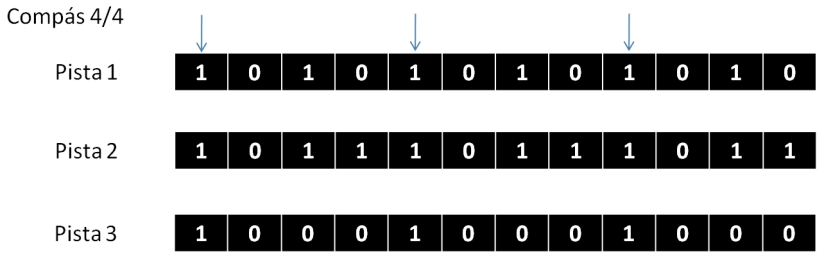

Figura 6.38: Ejemplo individuo obtenido en prueba 3

Se ve en la figura 6.38 como en este caso existen menos lugares donde se debe marcar el compás al ser cada cuatro posiciones en lugar de cada dos como ocurría en la prueba anterior. También se comprueba como sigue

cumpliéndose lo que la función de fitness aporta y salvo en esas posiciones, nunca coinciden más de dos notas en un mismo momento. Añadiendo la función de fitness vertical el sistema sigue creando ritmos adaptados a las configuraciones deseadas.

Prueba 4 (P=3, N=1000, C=2/4, M=0)

Como última prueba para esta función de fitness, se incrementa el número de notas por pista hasta las 1000 y se observa la evolución de la población guiada por el sistema:

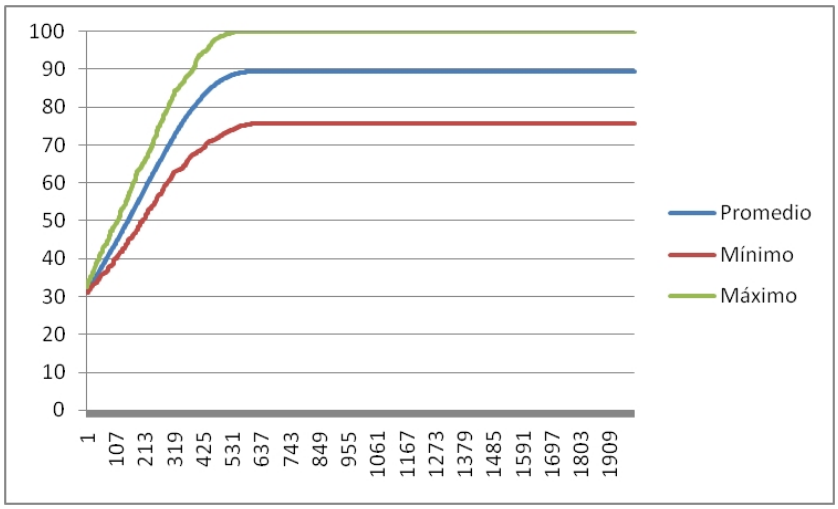

Figura 6.39: Evolución población con fitness principal, secundarios, ritmo y vertical en prueba 4

La figura 6.39 refleja que los resultados siguen siendo positivos y se alcanzan individuos con fitness muy elevados. A pesar de esto, el valor promedio obtenido se encuentra algo por debajo de los obtenidos en experimentos anteriores. Es ahora el primer momento en el que este valor se ve afectado en mayor grado con el aumento del número de notas por pista y se debe a que al aumentar las restricciones y los lugares donde se deben reproducir estas restricciones, le resulta más difícil al sistema satisfacerlas todas y en todos los lugares.

Conclusiones

Los número conseguidos con este conjunto de pruebas continúan con la tónica anterior y arrojan resultados muy positivos y esperanzadores. La población evoluciona de forma adecuada a las restricciones impuestas por las funciones de fitness y se generan cada vez individuos más complejos y configurables, adaptados a las características concretas de cada caso.

La variabilidad en las soluciones se sigue haciendo patente y las combinaciones que se pueden obtener son elevadas y variadas. Algo que comienza a notarse es que los individuos de la población se parecen mucho entre sí a medida que avanzan las generaciones, pero es un hecho que no supone ningún problema para la versión automática del sistema y que ha sido variado, como ya se ha comentado, para la versión interactiva.

Otro factor que vuelve a hacerse notar, de forma esperada y deseada, es que las puntuaciones otorgadas a cada tipo de fitness condicionarán de una manera enorme la evolución y, por consiguiente, los individuos resultantes, permitiendo aún más variabilidad condicionada por los gustos del usuario o por el tipo de ritmo que se desee conseguir en cada momento.

Como viene siendo habitual, se muestra a continuación una tabla resumen de los valores numéricos obtenidos en las pruebas seleccionadas:

	Mutación	Compás	Nº notas	Promedio	Mínimo	Máximo
Prueba 1	0,0	2/4	100	92,03383	80,56794	99,98727
Prueba 2	0,01	2/4	100	92,15098	74,97135	99,98727
Prueba 3	0,0	4/4	100	94,87001	78,88743	99,7788
Prueba 4	0,0	2/4	1000	89,49926	75,64592	99,84019

Cuadro 6.6: Resultados pruebas de fitness patrones principales, secundarios, ritmo y vertical

Se observa en la tabla 6.6 como los datos obtenidos no difieren demasiado de los de las funciones de fitness comentadas con anterioridad. Destaca el leve descenso del promedio alcanzado en la prueba 4. La explicación de ese descenso viene por el aumento en la complejidad de las individuos generados, que hacen más dificultoso para el sistema lograr ritmos que cumplan todos y cada uno de los puntos de manera exacta.

A pesar de eso, reiterar que el no alcanzar porcentajes cercanos al 100 % no tiene porque indicar que los individuos sean incorrectos. Los ritmos que se han generado siempre son aceptables y cumplen en su gran mayoría los requisitos, pero teniendo variedad en las puntuaciones otorgadas y permitiendo cierta originalidad y variabilidad en los resultados, se hace en ocasiones imposible obtener puntuaciones cercanas a la perfección, aunque las composiciones suenen correctamente.

En la sección siguiente se comentará la última función de fitness implementada y que se corresponde con aquella que regirá el funcionamiento de las pistas con un sonido correspondiente al instrumento del bajo.

6.2.4. Fitness orientado al bajo

Esta sección dará cabida a las diferentes pruebas realizadas añadiendo esta función de fitness a las ya relatadas con anterioridad. Esta función se aplica únicamente a aquellas pistas en las que se reproduzca un sonido correspondiente al bajo y su funcionamiento consiste en seguir al bombo o en caso de que no exista un bombo en la composición, a una de las pistas que reproduzcan sonidos de la batería. Esta regla viene creada de esta forma debido a que el sonido del bajo suele acompañar al bombo de la batería de manera que se reproduzcan simultáneamente.

Por tanto esta función de fitness valorará como positivo el hecho de que al producirse un sonido del bombo, se produzca también un sonido de bajo. La puntuación que se ha otorgado a este fitness es de 20 por cada posición del array de notas que satisfaga este requisito. No habrá penalización para aquellas posiciones que no lo cumplan. El resto de puntuaciones continuarán como hasta ahora.

Se ha pensado adecuado este valor debido a que debe representar un peso importante en el proceso creativo, pero no debe de convertirse en un factor determinante para la propia creación de la composición, dejando también cierta libertad a los resultados generados y que ambas pistas no sean siempre exactamente iguales, dando lugar a fragmentos de canción en que ambos puedan tocar separados y produzcan combinaciones algo más creativas.

Una vez explicados los detalles correspondientes a la nueva función de fitness introducida, se reflejan los resultados obtenidos tras la ejecución del conjunto de pruebas seleccionado.

Prueba 1 (P=3, N=100, C=2/4, M=0)

Se ejecuta el sistema con la primera de las configuraciones establecidas compuesta por 3 pistas de 100 notas cada una, con compás de 2/4 y un valor de mutación nulo.

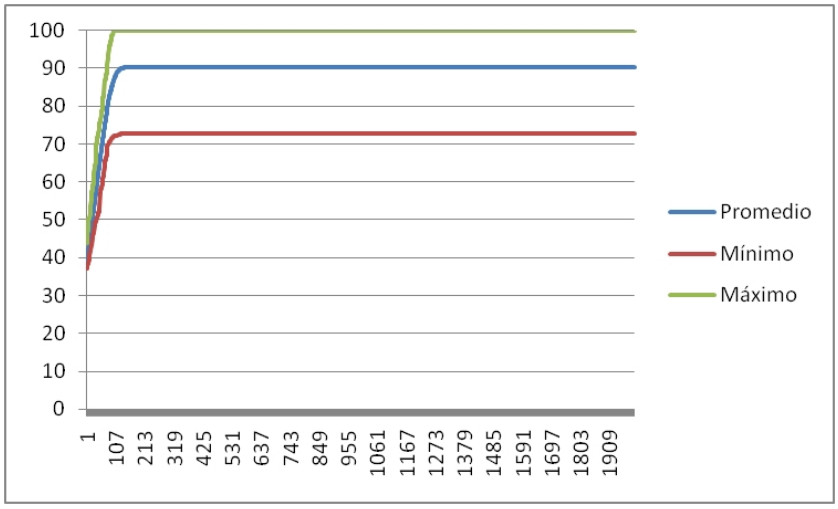

Figura 6.40: Evolución población con fitness principal, secundarios, ritmo, vertical y bajo en prueba 1

Los resultados obtenidos y recogidos en la figura 6.40 son tan positivos como los estudiados a lo largo de las pruebas anteriores. La evolución continúa siendo muy rápida. Se consiguen individuos cercanos al 100 % del fitness. Estos resultados concuerdan con los esperado, ya que esta nueva función no añade excesiva complejidad al sistema.

Las figuras 6.41 y 6.42 muestran dos de los individuos generados tras la evolución de la población empleando esta función de fitness:

Bombo	1	1	1	1	1	1	1	1	1	1	1	1
Bajo	1	0	1	0	1	0	1	0	1	0	1	0
Batería	0	0	1	0	0	0	1	0	0	0	1	0

Figura 6.41: Ejemplo 1, individuo obtenido en prueba 1

Este primer ejemplo muestra una situación en la que el bajo únicamente sigue al bombo en la mitad de las posiciones, pero sin embargo sigue siempre el mismo patrón. Se observa claramente como para este caso ha pesado más el seguir los patrones que el propio movimiento del bombo. El siguiente ejemplo muestra un resultado diferente:

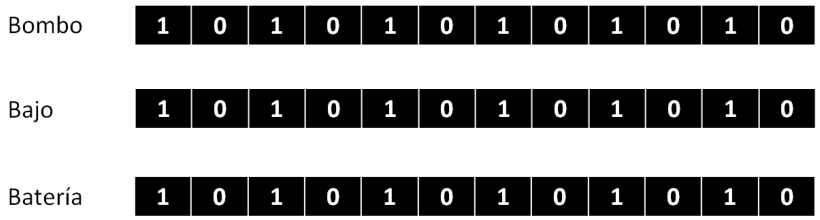

Figura 6.42: Ejemplo 2, individuo obtenido en prueba 1

Para el caso de la figura 6.42 el bajo sigue exactamente el patrón marcado por el bombo. Además el patrón seleccionado es el que otorga un fitness máximo al cumplir todas las características que definen las funciones de fitness. Independientemente de esto, es claro el hecho de que el sistema continúa adaptándose a los cambios y adecuando los resultados generados a cada configuración y situación concreta.

Las siguientes pruebas proporcionarán información adicional sobre la nueva función de fitness incluida.

Prueba 2 (P=3, N=100, C=2/4, M=0,01)

A continuación se ejecutará la segunda prueba estándar que mantiene la configuración de la prueba anterior pero el valor de la mutación cambia a 0,01. La figura 6.43 refleja los resultados obtenidos por esta configuración:

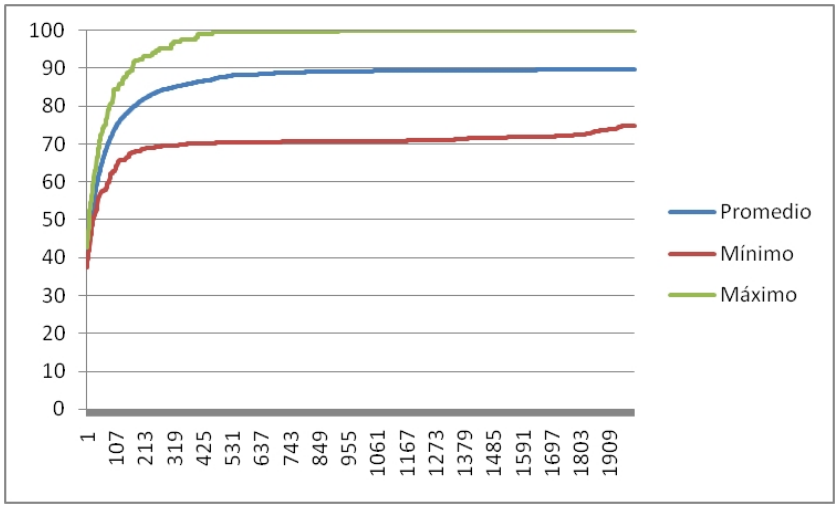

Figura 6.43: Evolución población con fitness principal, secundarios, ritmo, vertical y bajo en prueba 2

Como se puede observar en la figura 6.43, los resultados obtenidos son muy similares a los de la prueba anterior, lo que indica que la mutación sigue sin perjudicar al sistema para estos valores de número de notas. En este caso incluso podría ser más lógico obtener mejores resultados que sin mutación debido a que al valorarse de forma positiva que el bajo siga al bombo y, por tanto, derive en dos pistas iguales, la mutación puede resultar beneficiosa para encontrar esos valores donde coincida con el bombo y así localice el patrón que debe seguir más fácilmente.

Prueba 3 (P=3, N=100, C=4/4, M=0)

La prueba 3 modifica la configuración anterior estableciendo un nuevo tipo de compás. El compás utilizado será el 4/4. También se vuelve a un valor nulo de mutación. La figura 6.44 muestra la evolución de la población con los parámetros comentados:

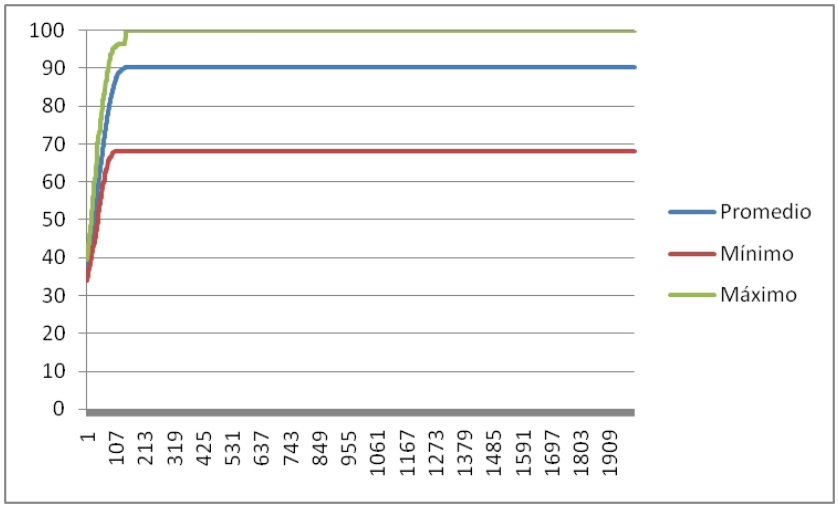

Figura 6.44: Evolución población con fitness principal, secundarios, ritmo, vertical y bajo en prueba 3

Los resultados extraídos de las pruebas y recogidos en la figura 6.44 muestran porcentajes muy parecidos a los de las pruebas anteriores. Para este caso parece que el compás no afecta de una forma tan clara como ocurría para las funciones de fitness estudiadas en el apartado anterior. Esto puede ser debido a que el peso adquirido por la función de fitness del bajo, hace que se atenúen las diferencias que existían en el pasado.

Prueba 4 (P=3, N=1000, C=2/4, M=0)

Esta prueba ejecutará la misma configuración que la prueba 1 pero aumentando de forma importante el número de notas que poseerá cada pista con el fin de comprobar si el sistema continúa con la capacidad de realizar sus tareas de manera adecuada y adaptarse a la situación. El número de notas por pista será de 1000. Los resultados se muestran en la figura 6.45:

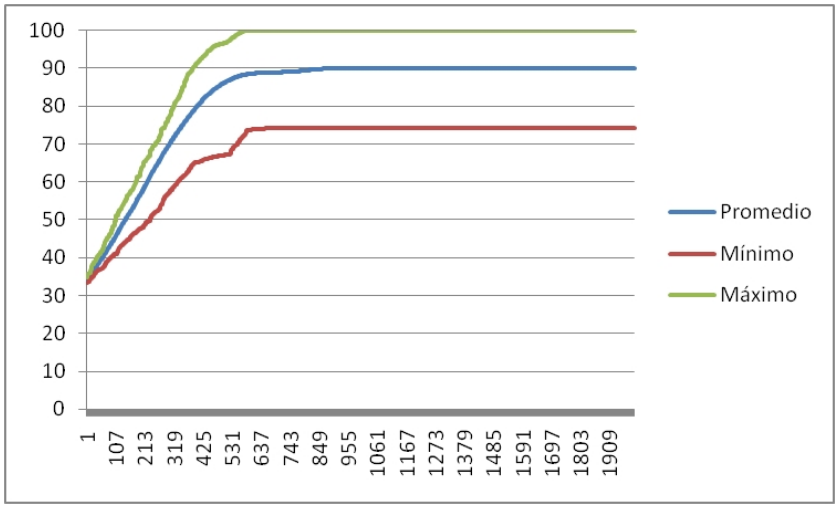

Figura 6.45: Evolución población con fitness principal, secundarios, ritmo, vertical y bajo en prueba 4

Es sencillo comprobar que los resultados obtenidos son idénticos a los extraídos con un número menor de notas. Esto viene a confirmar la adaptación del sistema para este tipo de condiciones y a que a pesar de incorporar cada vez mayor complejidad y variables de cálculo, sigue respondiendo de forma positiva y evolucionando los individuos hasta conseguir que se consoliden como válidos para ese entorno.

Prueba 5 (P=3, N=1000, C=2/4, M=0,01)

La última prueba que se realizará tiene como objetivo el analizar el efecto de la mutación sobre el conjunto de todas las funciones de fitness desarrolladas. Para ello, se continúa con un valor de 1000 notas por pista y se actualiza la mutación a un valor de 0,01 % del total de la población. La figura 6.46 muestra los resultados obtenidos por el sistema:

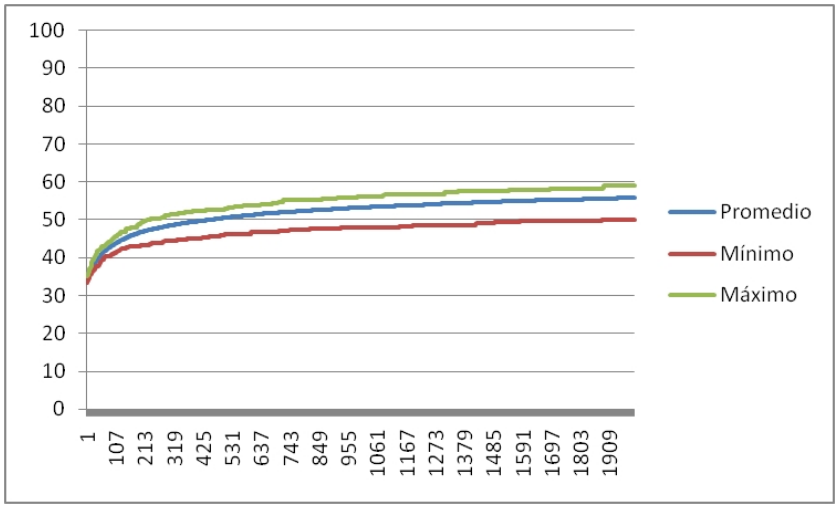

Figura 6.46: Evolución población con fitness principal, secundarios, ritmo, vertical y bajo en prueba 5

Se comprueba en la figura 6.46 como a pesar de las diversas configuraciones probadas a lo largo de este capítulo, para valores elevados de notas, la existencia de la mutación es altamente perjudicial. Por esta razón se recomienda adaptar los valores de la misma a las circunstancias concretas de la ejecución con el objetivo de optimizar los resultados. El conjunto de experimentos detallado aquí puede servir de guía para orientar los valores.

Conclusiones

Los resultados obtenidos con el conjunto de los experimentos realizados continúan siendo muy positivos. Las poblaciones evolucionan según lo previsto y se consiguen soluciones diversas y adecuadas en cada caso, teniendo siempre en cuenta las diversas restricciones impuestas por las funciones de fitness.

La variabilidad de las soluciones continúa siendo elevada y es uno de los factores críticos para el sistema y requisito inamovible. La diversidad de los individuos generados es uno de los pilares básicos sobre los que se asienta el sistema para poder crear composiciones distinguibles, agradables al oído e incluso capaces de inspirar melodías más complejas.

En relación a esto y observando los individuos generados en cada una de las pruebas realizadas, se observa que el hecho de restringir de esta manera que una pista sea igual a otra hace perder ese carácter variado y en un

principio aleatorio del sistema. Además de este problema, se une el hecho de que una pista sólo reproduce una nota y el hecho de que siempre suene la misma nota del bajo con el bombo puede resultar repetitivo o monótono. Por ello, tras ejecutar las pruebas y obtener los resultados se ha decidido prescindir de esta función de fitness para composiciones posteriores, aunque dejando el camino abierto para posibles desarrollos futuros que solventen los problemas planteados y otorguen la diversidad y calidad deseados.

La tabla 6.7 muestra los resultados extraídos de cada uno de los experimentos detallados con anterioridad:

	Mutación	Compás	Nº notas	Promedio	Mínimo	Máximo
Prueba 1	0,0	2/4	100	90,32625	72,73805	99,9887
Prueba 2	0,01	2/4	100	89,75545	74,73,173	99,9887
Prueba 3	0,0	4/4	100	90,43597	67,9572	99,80994
Prueba 4	0,0	2/4	1000	90,05	74,30222	99,99944
Prueba 5	0,01	2/4	1000	55,80518	49,93223	59,10585

Cuadro 6.7: Resultados pruebas de fitness patrones principales, secundarios, ritmo, vertical y bajo

Como se puede comprobar en la tabla 6.7, todos los valores continúan moviéndose en torno a los mismos porcentajes que en casos anteriores, siendo en general muy positivos y abarcando un abanico muy amplio en las soluciones que van desde el mínimo al máximo.

Llama la atención sin embargo la diferencia existente entre los valores mínimos obtenidos en las pruebas. Para la prueba 2 es muy probable que la subida del valor mínimo con respecto a los otros se deba al efecto de la mutación a la hora de encontrar una pista igual a la del bombo, que haga variar los patrones obtenidos mediante el cruce y se pueda adaptar mejor a esa pista. En el caso de la prueba 3 podría ocurrir justo el caso contrario. Al ser un compás de mayor longitud y no existir mutación, le resulta mucho más complicado adaptarse a la pista correspondiente al bombo empleando sólo el cruce por compás. El caso de la prueba 1 se sitúa en medio debido a que la longitud del compás no es tan larga como en la prueba 3, pero

no posee el pequeño porcentaje de mutación que facilita la situación en la prueba 2.

Como conclusión final destacar la calidad de los individuos generados, a gran adaptación del sistema a diversas configuraciones o entornos, la rapidez con que se obtienen resultados positivos y sobre todo la gran variedad en las soluciones generadas, que hacen del sistema una opción factible para componer ritmos ordenados de manera automática.

6.3. Sistema de creación interactivo

Como se ha comentado anteriormente, las pruebas para la versión interactiva del sistema resultan inviables. Los motivos de esta conclusión es que el objetivo de la versión interactiva del sistema es que el usuario dirija la evolución del mismo en relación a sus gustos personales o al tipo de composición que quiera conseguir en cada caso.

Por tanto, esa subjetividad que envuelve los resultados hace que las composiciones que puedan resultar agradables para unos, puedan ser poco interesantes para otros.

Capítulo 7

Conclusiones

La música, como todo arte, lleva asociado una gran carga de complejidad y subjetividad. Existen numerosas variables y variaciones que pueden hacer que un mismo conjunto de sonidos produzca sensaciones muy diferentes. Sin embargo, una composición compleja no tiene por qué ser mejor que una más sencilla. Prueba de ello es el tema de Coldplay "Viva la vida", gran éxito a nivel mundial cuya melodía consiste en la repetición de únicamente tres acordes. Con esta reflexión se pretende dar idea de esa complejidad implícita en la música y de la subjetividad que acompaña a cada uno de sus aspectos.

Con este proyecto se ha afrontado la composición musical de una "reducida" parte del conjunto de las posibilidades que ofrece la música, desde una perspectiva diferente a la convencional. La utilización de la Inteligencia Artificial y más concretamente de la computación evolutiva es un hecho que se ha venido produciendo desde los últimos años, pero este proyecto presenta diferencias notables con los existentes hasta la fecha. Para empezar es el primer sistema existente cuyos sonidos son del tipo ".wav" con las dificultades que eso conlleva. Además el planteamiento del problema y las versiones interactiva y automática del sistema suponen también una línea algo separada de lo previamente desarrollado. Todos estos detalles inyectan un valor añadido a los resultados, pues no existe base alguna sobre la que compararlos y supone el inicio de una vertiente no explorada hasta el momento.

Los primeros pasos en el diseño y desarrollo del sistema fueron complicados. Las dificultades tecnológicas encontradas desde el comienzo ralentizaron el proceso. Las escasas opciones de diseño e implementación y la complejidad de cada una de ellas, hicieron de los estudios preliminares un camino tedioso en el que se tuvieron que explorar y experimentar cada una de las tecnologías con el fin de conocer exactamente las prestaciones que ofrecían

así como comprobar cual de ellas se adaptaba mejor para lograr los objetivos planteados.

Tras esta incursión en las tecnologías existentes para manejar sonidos, se tomó la decisión que hoy se sigue considerando como la más acertada y se realizó el diseño del sistema sin perder nunca de vista aquello que se quería alcanzar. Entonces surgieron las dos vertientes o versiones del sistema: la automática y la interactiva. La primera de ellas pretendía generar de manera rápida y eficaz, bases rítmicas que se adecuaran a unas pautas sin más interacción con el usuario que seleccionar unas pocas características que debía cumplir la composición generada. La segunda iba un paso más allá y proponía al usuario una serie de composiciones para que éste eligiese cuál de ellas eran más de su agrado, de forma que el sistema pudiese evolucionar las canciones en ese sentido. Al hilo de estas dos versiones surgió también una tercera que ejercería de reproductor de canciones que previamente habían sido guardadas. Para poder navegar entre las diferentes opciones propuestas, se desarrolló un interfaz sencillo que ayudaría al usuario a manejar el sistema de una forma fácil y cómoda.

El desarrollo del algoritmo genético y de las diversas funciones de fitness otorgaron al sistema la capacidad de mejorar progresivamente las composiciones generadas. Esta fase fue la más "creativa" e interesante debido a la gran cantidad de posibilidades que se ofrecían. Se pretendía proporcionar la suficiente información al sistema como para crear bases rítmicas válidas y con sentido, sin que por ello se limitase en exceso la variedad de las composiciones generadas.

Todos los experimentos realizados comenzaron empleando únicamente sonidos de batería para, posteriormente, aumentar las posibilidades incluyendo también el bajo. Su elección se debió a su afinidad con los sonidos de batería y a que su uso conjunto está ampliamente implantado en la música. A partir de ese momento se desarrolló la función de fitness para el bajo que, como se comentó en anteriormente, fue eliminada por restringir demasiado las soluciones y eliminar parte del carácter aleatorio y creativo del sistema.

Es importante destacar el valor de todos y cada uno de los parámetros que componen el sistema ya que todos influyen en mayor o menor medida en las composiciones generadas. La duración de la canción, el compás, los instrumentos, los pesos de cada función de fitness... son determinantes para que una población evolucione en un sentido u otro.

Los resultados obtenidos para todas las versiones del sistema han sido más que satisfactorios. Los diversos y numerosos experimentos realizados han permitido estudiar ampliamente las posibilidades del sistema así como confirmar el buen funcionamiento del mismo. Se han cumplido, por tanto, los

objetivos marcados desde el inicio y las salidas generadas permiten observar las diferencias existentes entre las diferentes configuraciones aplicadas. Las bases rítmicas originadas son muy variadas y se obtienen desde ritmos clásicos que podrían servir como base para una canción pop hasta otras que con un acompañamiento vocal podrían escucharse en cualquier discoteca. Esto otorga unas posibilidades enormes en muchos sentidos y es un campo que debe seguir explorándose para poder alcanzar objetivos cada vez mayores.

Como conclusión final y opinión personal del autor decir que aunque el desarrollo de este proyecto ha sido largo y complicado, los resultados obtenidos y, sobre todo, las conclusiones, innovaciones y líneas de investigación que se han establecido con este sistema, han merecido el esfuerzo invertido y las intensas horas de trabajo dedicadas. Se puede concluir por tanto, que este sistema posee un gran valor que se encuentra ligado a los resultados que genera, pero también íntimamente relacionado con la labor experimental y de investigación que ha supuesto, indicando nuevos caminos por los que avanzar en este campo y que ojalá puedan seguir siendo explorados en un futuro cercano.

Capítulo 8

Líneas futuras

Como se ha comentado en varios puntos a lo largo de este documento, este proyecto deja varias líneas de investigación abiertas para el futuro. Estas líneas pueden ir encaminadas tanto a la mejora de las diversas variaciones del algoritmo presentado, como a añadir funcionalidades y explorar nuevos campos de creación musical.

Dentro de estas posibilidades se presentan las siguientes líneas de trabajo:

- El mundo de la música ofrece un número de posibilidades casi ilimitado, por lo que sería posible definir más funciones de fitness que tratasen de alcanzar los objetivos centrándose en otros aspectos, al margen de los presentados en este proyecto.
- Añadir diferentes tipos de cruce o mutación y probar las combinaciones entre ambos para tratar de mejorar los resultados obtenidos.
- Otorgar la posibilidad de seleccionar un sonido concreto para una pista, dando la opción al usuario de escuchar el sonido previamente y poder así buscar un resultado de la composición más ajustado a lo que se desea aunque perdiendo la originalidad y aleatoriedad perseguidas por este proyecto.
- Incorporar la funcionalidad ya mencionada de añadir efectos musicales a las notas. De este modo se consigue un abanico muy amplio de timbres que permiten jugar con las diversas posibilidades que ofrece el sonido.
- Crear un interfaz de usuario más amigable y vistoso, sin que por ello deje de un lado la sencillez y la facilidad de uso de la actual.

- Incorporar estilos musicales, es decir, la posibilidad de que en función del estilo musical que desee el usuario, se realicen composiciones que se adapten al estilo en cuestión.
- Posibilidad de que el usuario incorpore su propia biblioteca de sonidos.
- Exportar las composiciones generadas a otros formatos de sonido, como por ejemplo *mp3*.

Bibliografía

[Biles, 1994] Biles, J. A. (1994). Genjam: A genetic algorithm for generating jazz solos. pages 131–137.

[Dahlstedt, 2001] Dahlstedt, P. (2001). A mutasynth in parameter space: Interactive composition through evolution. *Organized Sound*, 6:121–124.

[Darwin, 1845] Darwin, C. (1845). *Journal of Researches into the Geology and Natural History of the Countries visited During the Voyage of H.M.S. Beagle Round the World (2nd ed.)*. London: Murray.

[Darwin, 1859] Darwin, C. (1859). *On the origin of species, by means of natural selection*. London: John Murray, Albemarle Street, London.

[de la Puente et al., 2002] de la Puente, A. O., Alfonso, R. S., and Moreno, M. A. (2002). Automatic composition of music by means of grammatical evolution. *SIGAPL APL Quote Quad*, 32(4):148–155.

[Fahlman and Lebiere, 1989] Fahlman, S. E. and Lebiere, C. (1989). The cascade-correlation learning architecture. In *NIPS*, pages 524–532.

[Fogel, 1994] Fogel, D. B. (1994). An introduction to simulated evolutionary optimization. *IEEE transactions on neural networks*, 5(1):3–14.

[Gibson and Byrne, 1991] Gibson, P. M. and Byrne, J. A. (1991). NEUROGEN: musical composition using genetic algorithms and cooperating neural networks. In *Second International Conference on Artificial Neural Networks, London*, pages 309–313. IEE.

[Griffiths et al., 2002] Griffiths, A., Miller, J., Suzuki, D., Lewontin, R., and Gelbart, W. (2002). *Genética, Séptima Edición*. McGraw-Hill, Madrid, ES.

[Grossman et al., 1993] Grossman, R. L., Nerode, A., Ravn, A. P., and Rischel, H., editors (1993). *Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*. Springer.

[Harmon and King, 1985] Harmon, P. and King, D. (1985). *Expert systems: artificial intelligence in business*. John Wiley & Sons, Inc., New York, NY, USA.

[Holland, 1975] Holland, J. H. (1975). *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA.

[Horner and Goldberg, 1991] Horner, A. and Goldberg, D. E. (1991). Genetic algorithms and computer-assisted music composition. In *ICGA*, pages 437–441.

[Horowitz, 1994] Horowitz, D. (1994). Generating rhythms with genetic algorithms. In *AAAI*, page 1459.

[Jacob, 1995] Jacob, B. L. (1995). Composing with genetic algorithms. In *International Computer Music Association*, pages 452–455.

[John A. Biles, 1996] John A. Biles, Peter G. Anderson, L. W. L. (1996). Neural network fitness functions for an iga. In *In Proceedings of the International ICSC Symposium on Intelligent Industrial Automation (ISA'96) and Soft Computing (SOCO'96), March 26–28*. ICSC Academic Press.

[Khalifa et al., 2007] Khalifa, Y. M. A., Khan, B. K., Begovic, J., Wisdom, A., and Wheeler, A. M. (2007). Evolutionary music composer integrating formal grammar. In *GECCO '07: Proceedings of the 2007 GECCO conference companion on Genetic and evolutionary computation*, pages 2519–2526, New York, NY, USA. ACM.

[Kurzweil, 1998] Kurzweil, R. (1998). *The age of spiritual machines: When computers exceed human intelligence*. Penguin group, New York, USA.

[Lovell, 1983] Lovell, M. C. (1983). Data mining. *The Review of Economics and Statistics*, 65(1):1–12.

[Miranda, 2001] Miranda, E. R. (2001). Music Technology Series. Focal Press, first edition. Book.

[Miranda et al., 2007] Miranda, E. R., Biles, J. A., and Miranda, E. R. (2007). *Evolutionary Computer Music*. Springer.

[Mitchell, 1996] Mitchell, M. (1996). *An introduction to genetic algorithms*. MIT Press, Cambridge, MA, USA.

[Moura, 2004] Moura, L. (2004). *Man + Robots : Symbiotic Art*.

[Moura, 2005] Moura, L. (2005). *Man + Robots : Symbiotic Art Bioart - A new Kind of Art*. Prates Gallery.

[Oliwa, 2008] Oliwa, T. M. (2008). Genetic algorithms and the abc music notation language for rock music composition. In *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 1603–1610, New York, NY, USA. ACM.

[Punch, 1993] Punch, W. F. (1993). Further research on feature selection and classification using genetic algorithms. In Forrest, S., editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo, CA. Morgan Kaufman.

[Ray, 1992] Ray, T. S. (1992). J'ai joué à dieu et créé la vie dans mon ordinateur. *Le Temps stratégique*, (47):68–81.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

[Shibata and Jennings, 1996] Shibata, T. Inoue, K. I. R. W. and Jennings, N. R., editors (1996). *Emotional robot for intelligent system-artificial emotionalcreature project*. IEEE.

[Vignoli, 2004] Vignoli, F. (2004). Digital music interaction concepts: A user study. In *ISMIR*.

[Wooldridge and Jennings, 1995] Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents: Theory and practice. *Knowledge Engineering Review*, 10:115–152.

[Zadeh, 1965] Zadeh, L. A. (1965). Fuzzy sets. *Information and Control*, 8(3):338–353.